Commonsense Reasoning and Large Network Analysis: A Computational Study of ConceptNet 4

Dimitrios I. Diochnos
Department of Mathematics, Statistics, and Computer Science
University of Illinois at Chicago, Chicago IL 60607, USA

Abstract

In this report a computational study of `ConceptNet 4` is performed using tools from the field of network analysis. Part I describes the process of extracting the data from the SQL database that is available online, as well as how the closure of the input among the assertions in the English language is computed. This part also performs a validation of the input as well as checks for the consistency of the entire database. Part II investigates the structural properties of `ConceptNet 4`. Different graphs are induced from the knowledge base by fixing different parameters. The degrees and the degree distributions are examined, the number and sizes of connected components, the transitivity and clustering coefficient, the cores, information related to shortest paths in the graphs, and cliques. Part III investigates non-overlapping, as well as overlapping communities that are found in `ConceptNet 4`. Finally, Part IV describes an investigation on rules.

Acknowledgements

The author would like to thank Robert H. Sloan and György Turán for suggesting him this subject. The author would also like to thank Tanya Y. Berger-Wolf for giving him access to a computer with sufficiently enough memory to perform the computations required in Chapter 10.

Contents

Part I

Closure of the Input, Validity, and Consistency

Chapter 1

Validity and Closure of the Database

Our aim is to compute the minimal data-set implied by the assertions of the English language, extract it from the database, and store it in files of our own format. Towards this direction we read the table of assertions (conceptnet_assertion) and keep the entries that have their `language_id` set to `en`. According to Table A.1 in Appendix A, every assertion is associated with entries from the database tables conceptnet_concept (Table A.2), conceptnet_relation (Table A.3), nl_frequency (Table A.4), conceptnet_frame (Table A.5), conceptnet_surfaceform (Table A.6), and conceptnet_rawassertion (Table A.7). Through conceptnet_rawassertion the assertions are also associated with the actual sentences which are located in the table corpus_sentence (Table A.6). Moreover, we do not need any other table from the database, as the important entries from all the above tables are contained in among these tables.

It turns out that reading once the assertions and then all the entries referenced from the assertions in the English language is not enough to produce a minimal consistent data-set. Section 1.1 explains why, and gives a high-level overview of the process that we follow in order to compute the closure of the data-set implied by the assertions of the English language. However, before we describe these reasons we mention which fields we are going to keep from each table of the original `ConceptNet 4` database.

Fields Retained from Each Database Table. A description of the information of the fields that we retain in every case follows.

conceptnet_assertion: Everything but the `language_id` field.

conceptnet_concept: `id`, `text`.

conceptnet_relation: Everything.

nl_frequency: Everything but the `language_id` field.

conceptnet_frame: The fields `question_yn`, `question1`, and `question2` are null in the entire database, hence we can safely ignore them. We are also dropping the `language_id` field as well as the `goodness` field.

conceptnet_surfaceform: We retain the information of the fields `id`, `concept_id`, and `text`.

conceptnet_rawassertion: We retain the information of the fields `id`, `sentence_id`, `assertion_id`, `surface1_id`, `surface2_id`, `frame_id`, and `score`.

corpus_sentence: We retain the information of the fields `id`, `text`, and `score`.

1.1 High Level Description for the Computation of the Closure

In this section we give a high-level description of the process that we follow in order to compute the closure of the data-set implied by the assertions of the English language. Ultimately we also want to use *igraph* [4] to aid our analysis of the networks induced by `ConceptNet 4`.

1.1.1 First Pass

During the first pass of the tables in the database we read the IDs of all the objects and store in matrices which IDs actually appear in the database. This is an important step since some references in the database are inconsistent. For example, some `best_raw_id`'s found in the table conceptnet_assertion point essentially to nowhere, since these particular IDs do not appear anywhere in the conceptnet_rawassertion table.

1.1.2 Second Pass

During the second pass of the tables in the database we extract the entries of assertions in the English language and all the entries from the other tables that are referenced from assertions. Moreover, we also extract all the sentences that are indirectly referenced by the assertions through the raw assertions. Ideally, one would expect that this process is enough in order to compute the minimal closure implied by the assertions in the English language. However, this is not the case. Below we describe the issues that arise after the first pass.

Null Entries. Some fields in the tables of the database do not have data associated to them. In the case of assertions in the English language these entries can appear in the fields `best_frame_id`, `best_surface1_id`, `best_surface2_id`, and `best_raw_id`. The assertion with the minimum ID that has `best_frame_id` equal to null is 344873. The assertion with the minimum ID that has `best_surface1_id` and `best_surface2_id` equal to null is 885221. The assertion with the minimum ID that has `best_raw_id` equal to null is 320114.

Undefined Raw Assertion IDs. There is an inconsistency problem regarding the IDs for the raw assertions that are mentioned in some entries of the assertions table. It turns out that 39312 different `best_raw_id`'s are not defined in the table of raw assertions; i.e. the IDs do not appear in the conceptnet_rawassertion table. The assertion instance with the minimum ID is 962 which points to `best_raw_id` 965.

Duplicate Raw Assertion IDs. Multiple assertions may point to the same raw assertion. Hence, not only we have assertions that have their `best_raw_id` equal to null or undefined, but the map between assertions and raw assertions is actually a *surjection*.

Discrepancies due to Frames. When we are able to read the frames in the field `best_frame_id` for an assertion (see Table A.1) we would expect that the `relation_id` and `frequency_id` mentioned in the relevant entry of the conceptnet_frame table (Table A.5) agree with the entries found in the assertion. However, it turns out that this is not necessarily the case for both of these values. For more information see Section 1.3.

Discrepancies due to Surface Forms. When we are able to read the surface forms in the fields `best_surface1_id` and `best_surface2_id` for an assertion (see Table A.1) we would again expect that the `concept_id` mentioned in the relevant entry of the conceptnet_surfaceform table (Table A.6) agree with the respective `concept1_id` and `concept2_id` entries in the conceptnet_assertion table (Table A.1). However, it turns out that this is not necessarily the case. In fact, this time the nature of disagreement can be dual:

- the IDs for the concepts disagree and both IDs are mentioned in the assertions, or
- the IDs for the concepts disagree but the IDs coming from the conceptnet_surfaceform table are not mentioned among the assertions in the English language.

The second case of disagreement above forces us to perform a second pass through the data so that we can collect all the data for the 388 concept IDs that did not appear during our first pass from the assertions in the English language. For more information about the quantitative properties of the discrepancies see Section 1.4.

Discrepancies due to Raw Assertions. When we are able to read the raw assertion ID in the fields `best_raw_id` for an assertion (see Table A.1) we would again expect that the entries `surface1_id`, `surface2_id`, and `frame_id` mentioned in the relevant entries of the conceptnet_rawassertion table (Table A.7) agree with the `best_surface1_id`, `best_surface2_id`, and `best_frame_id` entries in the conceptnet_assertion table (Table A.1). However, it turns out that this is not necessarily the case either. Similarly to the case above where we find 388 concepts not mentioned among the assertions in the English language, this process uncovers 540 surface form IDs that were not best surfaces for any assertion in the English language. Moreover, note that from our earlier remark the map of assertions to raw assertions is a surjection, and hence it is guaranteed that there is a discrepancy in the `assertion_id` entry of the conceptnet_rawassertion table. For more information see Section 1.5.

Discrepancies on the Score Entries. Most of the assertions have a valid raw assertion associated to them. Moreover, every raw assertion is associated with an actual sentence. Inspecting the Tables A.1, A.7, and A.8 we see that each of the tables conceptnet_assertion, conceptnet_rawassertion, and corpus_sentence has an entry with the score associated score. One would expect that all these three scores actually agree with each other whenever

we have a valid chain of the form: assertion $\longrightarrow$ raw assertion $\longrightarrow$ sentence. However, it turns out that this is not true either. For more information see Section 1.6.

End of Second Pass At the end of second pass we can observe that 459662 assertions have all their indicators equal to zero, out of which 450205 have positive score and the rest 9457 have non-positive score. Given the fact that we have the mapping assertions $\twoheadrightarrow$ raw assertions we could allow the indicator for the raw assertions to achieve, apart from zero, the value 18 as well (see Table 1.3). However, the numbers mentioned above *do not change at all*.

1.1.3 Third Pass

In the third pass we parse the data in the tables conceptnet_concept and conceptnet_surfaceform. This allows us to load the concepts and the surface forms that were raised from the previous pass. In theory, it could be the case that these new additional surface forms were referring to concepts that have not been raised yet from the previous passes, and hence we would require one more pass on the conceptnet_concept table to add these last concepts. However, this is not the case. In other words, these newly introduced surface forms from the last pass do not refer to concepts that we have not encountered earlier. Hence, this third pass is the last pass that we perform on the tables of the database.

1.2 First Pass: Validating IDs

There is not much to be said about the first pass. We parse all 8 tables of the database and record which IDs of these objects are actually valid IDs in the sense that references from other tables to these objects are guaranteed to return a result. The issue that forces us to follow this direction is the fact that some `best_raw_id`'s found in the conceptnet_assertion table actually point to nowhere, since we can not find raw assertions with these specific IDs in the table conceptnet_rawassertion.

1.3 Second Pass: Discrepancies due to Frames

Looking at Tables A.1 and A.5 we would expect the relations and the frequencies mentioned in the associated entries to agree. However, this is not always true in both cases. Moreover, when we have discrepancies among the relations or the frequencies, these other values appear in some other assertion in the English language. We mention this because in Section 1.4 similar discrepancies will occur only among values that can be observed based on the input from the assertions in the English language.

Regarding the relations, there are 816 assertions that have `best_frame_id` equal to null. Among the not null entries, in 564445 assertions the relation ID from the conceptnet_assertion table agrees with the respective relation ID from the relevant entry in the conceptnet_frame table. The rest 833 assertions have relation ID different from the relevant entry mentioned in the table conceptnet_frame.

Regarding the frequencies, there are again 816 assertions that have `best_frame_id` equal to null. Among the not null entries, in 562798 assertions the frequency ID from the conceptnet_assertion table agrees with the respective frequency ID from the relevant entry in the conceptnet_frame table. The rest 2480 assertions have a frequency ID different from the relevant entry mentioned in the table conceptnet_frame.

Remark 1 (Interesting Phenomenon)**.** *These two fields never disagree simultaneously with the entries found in the conceptnet_assertion table. This implies that if the relation changes, then the frequency, which expresses the extent to which the relation holds, does not change. Moreover, if the frequency changes, i.e. the extent to which the relation holds, then the relation does not change.*

The above information is captured in Table 1.1.

Examples. We give one example for each case presented in Table 1.1.

Indicator = 0. Assertion 2 associates the concepts 5 (`something`) and 6 (`to`) with relation 6 (`AtLocation`) and frequency 1 (which has value $5 (> 0)$ and the empty string for description). The best raw assertion for this assertion has ID 3 which is associated with the sentence `Somewhere something can be is next to`. The

Table 1.1: Distribution of the frame indicator.

indicator	description	entries
0	`best_frame_id` is not null, relations agree, frequencies agree	564445
1	`best_frame_id` is not null, relations agree, frequencies disagree	2480
2	`best_frame_id` is not null, relations disagree, frequencies agree	833
3	`best_frame_id` is not null, relations disagree, frequencies disagree	0
4	`best_frame_id` is null	816
	sum	566094

best frame for the assertion is 3 which is `Somewhere {1} can be is next {2}` and the relation associated to that frame is again 6 (`AtLocation`), as well as the frequency is 1.

Indicator = 1. Assertion 36294 associates the concepts 481 (`milk`) and 1503 (`refrigerator`) with relation 6 (`AtLocation`) and frequency 1 (which has value 5 and the empty string for description). The best raw assertion for this assertion has ID 368705 which is associated with the sentence `Something you find the refrigerator is milk.`. The best frame for the assertion is 2761 which is `Something you find {2} is {1}.`. However, this frame is associated with relation 6 (`AtLocation`) and frequency 25 (which has value $-5 < 0$ and the string description is `not`)!

Indicator = 2. Assertion 17691 associates the concepts 18845 (`bread knife`) and 13506 (`cut bread`) with relation 7 (`UsedFor`) and frequency 1 (which has value 5 and the empty string for description). The best raw assertion for this assertion has ID 18168 which is associated with the sentence `bread knives are for cutting bread`. The best frame for the assertion is 40 which is `{1} are {2}`. However, this frame is associated with frequency 1 and relation 5 (`IsA`)!

Indicator = 3. No instances.

Indicator = 4. Assertion 344873 associates the concepts 217239 (`don't`) and 217240 (`manipulate gene`) with relation 8 (`CapableOf`). However, the `best_frame_id` field for this assertion is null. The `best_raw_id` field is also null.

1.4 Second Pass: Discrepancies due to Surface Forms

Every assertion has two best surface forms; one for each concept. Hence, `best_surface1_id` describes the concept in the entry `concept1_id`, and `best_surface2_id` describes the concept `concept2_id`. On the other hand, every entry in the table conceptnet_surfaceform associates each surface form with a concept and also provides a string representation of that concept in that particular surface form.

Ideally, we would expect `concept1_id` or `concept2_id` from the table conceptnet_assertion to match respectively with the entry of `concept_id` of the respective surface ID (i.e. respectively `best_surface1_id` and `best_surface2_id`). However, it turns out that this is not the case.

First of all, the table conceptnet_assertion has 810 entries in the English language where the `best_surface1_id` and `best_surface2_id` are simultaneously null (all other entries in the English language do not have null entries for either of these two parameters). Moreover, even when we do have valid surface IDs in the respective entries of the table conceptnet_assertion, the `concept_id` on the relevant entry of the table conceptnet_surfaceform may point to a concept with ID not matching the respective one obtained through conceptnet_assertion. In fact, it may not match the relevant concept ID of the conceptnet_assertion table in two different ways:

- the concept IDs differ and both are part of the input, or
- the concept IDs differ but the concept with ID `concept_id` is *missing* (i.e. does not appear) among all the assertions of the English language.

The last case may happen either because that concept ID did not appear in any assertion but does appear in the conceptnet_concept table, or (I have not checked and I find quite unlikely the following) because no such concept ID appears in the conceptnet_concept table (similar phenomenon to that observed on the IDs of raw assertions).

As a consequence of the above we can distinguish 16 cases which are shown in Table 1.2.

Table 1.2: Distribution of the indicator for surface forms.

indicator	description	entries
0	surface 1 not null and concept IDs agree; surface 2 not null and concept IDs agree	561530
1	surface 1 not null and concept IDs agree; surface 2 not null and concept IDs disagree	2513
2	surface 1 not null and concept IDs agree; surface 2 not null and concept ID missing	383
3	surface 1 not null and concept IDs agree; surface 2 is null	0
4	surface 1 not null and concept IDs disagree; surface 2 not null and concept IDs agree	814
5	surface 1 not null and concept IDs disagree; surface 2 not null and concept IDs disagree	28
6	surface 1 not null and concept IDs disagree; surface 2 not null and concept ID missing	3
7	surface 1 not null and concept IDs disagree; surface 2 is null	0
8	surface 1 not null and concept ID missing; surface 2 not null and concept IDs agree	13
9	surface 1 not null and concept ID missing; surface 2 not null and concept IDs disagree	0
10	surface 1 not null and concept ID missing; surface 2 not null and concept ID missing	0
11	surface 1 not null and concept ID missing; surface 2 is null	0
12	surface 1 is null; surface 2 not null and concept IDs agree	0
13	surface 1 is null; surface 2 not null and concept IDs disagree	0
14	surface 1 is null; surface 2 not null and concept ID missing	0
15	surface 1 is null; surface 2 is null	810
	sum	566094

Examples. Below we give the first example (as we parse the assertions in order) for each case of the indicator variable for surface forms.

Indicator = 0. Assertion 2 relates concepts 5 (`something`) and 6 (`to`). The best surface forms have IDs respectively 5 and 6 (same numbers as the concept IDs; it just happened). These surface forms in turn point to the same concepts that we have in the assertion; i.e. 5 and 6 respectively and the text representation of the concepts obtained from the conceptnet_concept table is the same as the text representation obtained from the conceptnet_surfaceform table.

Note that the text representation of the concepts need not be the same in general, even in this class. An example in this direction is assertion 7 which relates concepts 13 (`strike match`) and 14 (`burn down church`). The best surface form IDs respectively are 14 and 15. The concept IDs obtained from these surface forms agree respectively with the concept IDs that we have in the assertion. However, the strings that we get through the surface forms are respectively `striking a match` and `burning down churches`.

Indicator = 1. Assertion 335 relates concepts 538 (`toothpaste`) and 327340 (`clean one tooth`). The best surface forms have IDs respectively 565 and 22753. The second surface form disagrees, since it points to the concept 311600. The string from the second surface form is `cleaning teeth`.

Indicator = 2. Assertion 29378 relates concepts 5 (`something`) and 312273 (`one with all that be`). The best surface forms have IDs respectively 5 and 186249. The concept obtained from the second surface form has ID 322557 and is not part of the input since it does not appear in any assertion in the English language. The string from the second surface form is `where it should be`.

Indicator = 3. No instances.

Indicator = 4. Assertion 1464 relates concepts 1906 (`most people`) and 1121 (`read book`). The best surface forms have IDs respectively 2173 and 2174. However, the concept obtained from the first surface form has ID 9. The string obtained for that concept from conceptnet_concept is `person` while the string obtained from the surface form is `most people`.

Indicator = 5. Assertion 6233 relates concepts 980 (`movies`) and 7356 (`show theater`). The best surface forms have IDs respectively 8520 and 8521. This time both concept IDs obtained from the surface forms disagree with the IDs of the concepts found in the assertion. The surface forms give respectively concept IDs 213 and 316392. The strings for these concepts from the conceptnet_concept table are respectively `movie` and

shown theater. The strings for these concepts from the surface form entries are respectively Movies and shown in theaters.

Indicator = 6. Assertion 280329 relates concepts 17626 (entertain people) and 186703 (make keep friends). The best surface forms have IDs respectively 36638 and 250423. The concept IDs obtained from the surface forms are respectively 427797 and 326698. The strings for these concepts from the conceptnet_concept table are entertain person and make keep friend. The strings for these concepts from the surface form entries are respectively entertaining people and making and keeping friends.

Indicator = 7. No instances.

Indicator = 8. Assertion 60579 relates concepts 49223 (wah one's hair) and 2697 (good idea). The best surface forms have IDs respectively 63422 and 63423. The concept IDs obtained from the surface forms are respectively 314140 and 2697. The strings for these concepts from the conceptnet_concept table are wah hair and good idea. The strings for these concepts from the surface form entries are respectively Wahing one's hair and is a good idea.

Indicator = 9, . . . , 14. No instances.

Indicator = 15. Assertion 885221 relates concepts 25036 (see particular program) and 643 (enjoyment). The best surface form IDs are null in both cases.

1.4.1 Concepts Raised

This verification process raises 388 concept IDs, all of which are valid, but were not mentioned among the assertions in the English language.

1.5 Second Pass: Discrepancies due to Raw Assertions

Table 1.3 shows the distribution for the indicator.

Examples. Below we give the first example (as we parse the assertions in order) for each case of the indicator variable for the raw assertions.

Indicator = 0. Assertion 2 has best raw assertion equal to 3 which is associated with the sentence Somewhere something can be is next to (ID 715991). The best frame for the assertion is 3 which is Somewhere {1} can be is next {2}. The best surface forms are respectively 5 (something) and 6 (to). The raw assertion points to the assertion 2 and has the same surface forms and frame.

Indicator = 1-17. No instances.

Indicator = 18. Assertion 674 has best raw assertion equal to 675 which is associated with the sentence something can be at the movies (ID 716856). Frame 43 ({1} can be at {2}) is the best frame for this assertion, and the two surface forms are respectively 5 (something) and 1047 (the movies). The raw assertion has the same frame and same surfaces respectively, but points to the assertion with ID 40199. !!! Interestingly enough, the assertion 40199 does not point back to this raw assertion but rather to the raw assertion with ID 43017 which is associated with the sentence Somewhere something can be is a movie.

Indicator = 19-27. No instances.

Indicator = 28. Assertion 7270 has best raw assertion equal to 7375 which is associated with the sentence speakers are for making sound (ID 728720). The best frame for this assertion is 40 ({1} are {2}) and the two surface forms are respectively 9819 (speakers) and 9820 (for making sound). The raw assertion has frame 7 ({1} is for {2}) and the surface forms are respectively 9819 (speakers) and 143185 (making sound) The raw assertion points to the assertion 429487 which has its best_raw_id equal to 7375. Both of these assertions, i.e. the one with ID 7270 and the one with ID 429487, relate the concepts 8419 (speaker) and 8420 (make sound).

Table 1.3: The distribution of the indicator for the discrepancies due to the raw assertions.

indicator	description				entries
0	assertion agrees,	frame agrees,	surface 1 agrees,	surface 2 agrees	523306
1	assertion agrees,	frame agrees,	surface 1 agrees,	surface 2 disagrees	0
2	assertion agrees,	frame agrees,	surface 1 agrees,	surface 2 missing	0
3	assertion agrees,	frame agrees,	surface 1 disagrees,	surface 2 agrees	0
4	assertion agrees,	frame agrees,	surface 1 disagrees,	surface 2 disagrees	0
5	assertion agrees,	frame agrees,	surface 1 disagrees,	surface 2 missing	0
6	assertion agrees,	frame agrees,	surface 1 missing,	surface 2 agrees	0
7	assertion agrees,	frame agrees,	surface 1 missing,	surface 2 disagrees	0
8	assertion agrees,	frame agrees,	surface 1 missing,	surface 2 missing	0
9	assertion agrees,	frame disagrees,	surface 1 agrees,	surface 2 agrees	0
10	assertion agrees,	frame disagrees,	surface 1 agrees,	surface 2 disagrees	0
11	assertion agrees,	frame disagrees,	surface 1 agrees,	surface 2 missing	0
12	assertion agrees,	frame disagrees,	surface 1 disagrees,	surface 2 agrees	0
13	assertion agrees,	frame disagrees,	surface 1 disagrees,	surface 2 disagrees	0
14	assertion agrees,	frame disagrees,	surface 1 disagrees,	surface 2 missing	0
15	assertion agrees,	frame disagrees,	surface 1 missing,	surface 2 agrees	0
16	assertion agrees,	frame disagrees,	surface 1 missing,	surface 2 disagrees	0
17	assertion agrees,	frame disagrees,	surface 1 missing,	surface 2 missing	0
18	assertion disagrees,	frame agrees,	surface 1 agrees,	surface 2 agrees	1848
19	assertion disagrees,	frame agrees,	surface 1 agrees,	surface 2 disagrees	0
20	assertion disagrees,	frame agrees,	surface 1 agrees,	surface 2 missing	0
21	assertion disagrees,	frame agrees,	surface 1 disagrees,	surface 2 agrees	0
22	assertion disagrees,	frame agrees,	surface 1 disagrees,	surface 2 disagrees	0
23	assertion disagrees,	frame agrees,	surface 1 disagrees,	surface 2 missing	0
24	assertion disagrees,	frame agrees,	surface 1 missing,	surface 2 agrees	0
25	assertion disagrees,	frame agrees,	surface 1 missing,	surface 2 disagrees	0
26	assertion disagrees,	frame agrees,	surface 1 missing,	surface 2 missing	0
27	assertion disagrees,	frame disagrees,	surface 1 agrees,	surface 2 agrees	0
28	assertion disagrees,	frame disagrees,	surface 1 agrees,	surface 2 disagrees	189
29	assertion disagrees,	frame disagrees,	surface 1 agrees,	surface 2 missing	607
30	assertion disagrees,	frame disagrees,	surface 1 disagrees,	surface 2 agrees	0
31	assertion disagrees,	frame disagrees,	surface 1 disagrees,	surface 2 disagrees	0
32	assertion disagrees,	frame disagrees,	surface 1 disagrees,	surface 2 missing	0
33	assertion disagrees,	frame disagrees,	surface 1 missing,	surface 2 agrees	0
34	assertion disagrees,	frame disagrees,	surface 1 missing,	surface 2 disagrees	0
35	assertion disagrees,	frame disagrees,	surface 1 missing,	surface 2 missing	0
				partial sum	525950
36	raw assertion is null				832
37	raw assertion is undefined				39312
				sum	566094

Indicator = 29. Assertion 29506 has best raw assertion equal to 31257 which is associated with the sentence `hands are for touching things` (ID 768019). The best frame for this assertion is 40 (`{1} are {2}`) and the two surface forms are respectively 2624 (`hands`) and 34422 (`for touching things`). The raw assertion has frame 7 (`{1} is for {2}`) and the surface forms are respectively 2624 (`hands`) and 287669 (`touching things`). The raw assertion points to the assertion 393267 which has its `best_raw_id` equal to 977445 which is associated with the sentence `hand is used for touch` (ID 2308541).

Indicator = 30-35. No instances.

Indicator = 36. The assertion with ID 320114 has `best_raw_id` equal to null.

Indicator = 37. The assertion with ID 962 points to an undefined `best_raw_id` (ID 965).

1.5.1 Surface Forms Raised

This verification process raises 540 surface form IDs, all of which are valid, but were not mentioned among the assertions in the English language.

1.6 Second Pass: Discrepancies on the Score Entries

Table 1.7 gives a distribution of discrepancies according to the metric h given by (1.2) in Section 1.6.3. Table 1.8 gives a distribution of discrepancies observed among the three tables that refer to scores.

1.6.1 Signs on Scores

Remark 2 (Two Signs for Scores)**.** *We distinguish only two signs for the scores. Strictly positive* (> 0) *and non-positive* ($\leqslant 0$)*. We do so, since every assertion when first entered into* `ConceptNet 4` *has score equal to* 1*. Hence, a non-positive score implies that the assertion is not so good. This approach was also followed in [13].*

Table 1.4 presents the number of entries that have positive and non-positive scores in the three tables.

Table 1.4: Positive and non-positive scores on the entries of the three tables. In the cases of the entries of raw assertions and sentences, the values are obtained by observing the entries in the chain assertion → `best_raw_id` → `sentence_id`.

	entries with positive score	entries with non-positive score	total entries
assertions	492389	73705	566094
raw assertions	493108	32842	525950
sentences	516324	9626	525950

1.6.2 Bounds on Scores

Tables 1.5 and 1.6 present the extreme values that scores can obtain in `ConceptNet 4`. Table 1.5 refers to the entire tables, while Table 1.6 refers to the entries that have their `language_id` equal to `en`.

Table 1.5: Bounds on scores from different tables; *any language*

	minimum score	maximum score	id for minimum score	id for maximum score
assertions	-10	311	330369	741038
raw assertions	-10	265	377317	566768
sentences	-10	265	1690862	1509374

Table 1.6: Bounds on scores from different tables when the language is restricted to *English*.

	minimum score	maximum score	id for minimum score	id for maximum score
assertions	-10	147	330369	1664
raw assertions	-10	124	377317	19218
sentences	-10	124/48	1690862	1241798/1318471

Minimum Score - Both Tables. The assertion with ID 330369 has `best_raw_id` equal to 377317, which in turn has `sentence_id` equal to 1690862. Hence, all the minimum values are obtained through the same sentence of the English language:

`college is a kind of musical instrument.`

The frame and the surfaces also agree with each other between the assertion and the raw assertion.

Maximum Score - Any language. The assertion with ID 741038 has score 311 and it is an assertion in Chinese (`language_id` is `zh-Hant`). It refers to the raw assertion with ID 981853 (score equal to 1), which in turn refers to the sentence with ID 2312949 (score equal to 1). Google Translate gives:

`You eat because you're hungry.`

The raw assertion with ID 566768 has score 265 and it is a raw assertion in Portuguese (`language_id` is `pt`). It refers to the sentence with ID 1897890 (score equal to 1). Google Translate gives:

`People sleep when they are sleepy.`

The sentence with ID 1509374 has score 265 and it is also a sentence in Portuguese (`language_id` is `pt`). According to Google Translate the sentence is:

`People sleep when they are sleepy`

Remark 3 (Slight Variations ⇒ Big Score Discrepancies)**.** *The last two examples in Portuguese differ only by a full stop! However, the difference in scores is very large.*

Maximum Score - English Language. The assertion with ID 1664 has score 147 and has `best_raw_id` 19218. It relates the concepts `baseball` (`concept1_id` is 1890) and `sport` (`concept2_id` is 2130) through the relation `IsA` (ID is 5).

However, the score for the raw assertion 19218 is 124 (i.e. different from 147). Note here, that this raw assertion points to the sentence with ID 748040 which is:

`Baseball is a sport played in the U.S.`

Regarding the maximum score obtained through the `corpus_sentence` table we have a very strange phenomenon. Just by looking at the table on those sentences that have a tag for the English language, the maximum score is 124 and is obtained through the sentence with ID 1241798. That sentence is:

`Baseball is a sport`

Remark 4 (Baseball inconsistency on Sentences)**.** *The sentence* `Baseball is a sport` *with ID* 1241798 *is* not *referred by* any *raw assertion! This is very strange, especially because the score of this sentence is* 124*, just like the score of the raw assertion with ID* 19218*, which refers to the sentence* `Baseball is a sport played in the U.S.` *which in turn has score only* 1 *(see above for the maximum score obtained among the raw assertions in the English language).*

On the other hand, if we look on all those sentences, that are associated with a raw assertion (i.e. we can find their IDs in some row of the `conceptnet_rawassertion` table), such that the raw assertion appears as `best_raw_id` in some assertion of the `conceptnet_assertion` table, then, the maximum score obtained is 48 through the sentence with ID 1318471 which is:

```
bottles are often made of plastic
```

We note here, that we have the same result even if we do the simpler search of finding the maximum score among the sentences in the English language that appear in some raw assertion of the English language. In other words, the following two sets of SQL queries return the same values:

- SQL Query Set 1:

```
sqlite> select max(score) from corpus_sentence where id in (
   ...> select sentence_id from conceptnet_rawassertion where id in (
   ...> select best_raw_id from conceptnet_assertion where language_id = 'en'));
48
sqlite> select id from corpus_sentence where score = 48 and id in (
   ...> select sentence_id from conceptnet_rawassertion where id in (
   ...> select best_raw_id from conceptnet_assertion where language_id = 'en'));
1318471
```

- SQL Query Set 2:

```
sqlite> select max(score) from corpus_sentence where id in (
        select sentence_id from conceptnet_rawassertion where language_id = 'en');
48
sqlite> select id from corpus_sentence where score = 48 and id in (
        select sentence_id from conceptnet_rawassertion where language_id = 'en');
1318471
```

1.6.3 Magnitude of Score Inconsistencies: Discrepancy and Half-Discrepancy

This section gives a brief description of the magnitude of the inconsistencies that can be observed as we restrict the assertions in the English language.

Definition 1 (Discrepancy)**.** *We define the discrepancy* d *to be*

$$d = |s_1 - s_2| + |s_2 - s_3| + |s_3 - s_1| \,, \tag{1.1}$$

where s_1, s_2*, and* s_3 *are the scores appearing respectively in the tables* `conceptnet_assertion`*,* `conceptnet_rawassertion`*, and* `corpus_sentence`*.*

Definition 2 (Half-Discrepancy)**.** *We define half-discrepancy to be*

$$h = \frac{d}{2} \,. \tag{1.2}$$

The following theorem guarantees that $h \in \mathbb{N}$, and hence d is an even natural number.

Proposition 1 (Integer Half-Discrepancies)**.** *Quantity* h *is a natural number.*

Proof. Let us look at the quantity $d = 2 \cdot h$ and note that $s_1, s_2, s_3 \in \mathbb{Z}$. We will prove that d can only be even. Towards contradiction, assume that d is odd. Then, d is either the sum of three odd values or one odd and two even values.

If d is the sum of one odd and two even values, then, without loss of generality we can assume that $|s_1 - s_2| = 2k + 1$, while $|s_2 - s_3| = 2m$ and $|s_3 - s_1| = 2n$, where $k, m, n \in \mathbb{N}$. Since $|s_2 - s_3| = 2m$ and $|s_3 - s_1| = 2n$, it follows that s_1 and s_2 have the same parity since they differ an even amount of integer values from s_3. However, this is a contradiction to $|s_1 - s_2| = 2k + 1$ which implies that the parity of s_1 and s_2 is different.

On the other hand, if d is the sum of three odd values, then without loss of generality we can assume that $|s_1 - s_2| = 2k + 1$, $|s_2 - s_3| = 2m + 1$, and $|s_3 - s_1| = 2n + 1$, where $k, m, n \in \mathbb{N}$. Similarly to the case above, since $|s_1 - s_2| = 2k + 1$ and $|s_2 - s_3| = 2m + 1$ it follows that s_1 and s_3 have the same parity because they differ an odd number of integer values from s_2. But this is a contradiction to the assumption that $|s_3 - s_1|$ is odd. ∎

Table 1.7 presents the distribution of the magnitude of the discrepancies that we can observe among the three tables that have score entries.

Table 1.7: Distribution of half-discrepancies h. Half-discrepancies are given by (1.2). In the cases where `best_raw_id` is null or undefined/missing, we set $d, h = 0$.

h	entries
0	504889
1	50972
2	5990
3	1976
4	931
5	499
6	224
7	122
8	51
9	143
10	86

h	entries
11	68
12	42
13	19
14	20
15	8
16	8
17	2
18	7
19	5
20	6
21	1

h	entries
22	3
23	2
24	1
25	2
26	1
29	1
32	1
33	2
35	1
36	1
39	1

h	entries
40	1
41	1
48	1
62	1
64	1
73	1
77	1
108	1
146	1

Regarding the instance that achieves the maximum discrepancy (146) please have a look in the discussion in Section 1.6.4; in particular when the indicator is equal to 8.

1.6.4 Enumerating Score Inconsistencies between Tables

Table 1.8 presents the inconsistencies among the score entries found in the three different tables as we read the assertions in the English language.

Table 1.8: Distribution of inconsistencies among the score entries found in the three different tables of `conceptnet_assertion`, `conceptnet_rawassertion`, and `corpus_sentence`. The last column presents the maximum half-discrepancy obtained in each group.

indicator	scores where …	entries	maximum h
0	all three agree	464745	0
1	`best_raw_id` is null or undefined/missing	40144	0
2	assertions and raw assertions agree; sentences have same sign	7614	15
3	assertions and raw assertions agree; sentences have different sign	22933	3
4	assertions and sentences agree; raw assertions have same sign	152	9
5	assertions and sentences agree; raw assertions have different sign	129	4
6	raw assertions and sentences agree; assertions have same sign	22915	73
7	raw assertions and sentences agree; assertions have different sign	1616	8
8	all three disagree; same sign (> 0, or $\leqslant 0$)	5670	146
9	all three disagree; different signs (> 0, or $\leqslant 0$)	176	15
	sum	566094	

Examples. We give one example for each case of the score discrepancy indicator.

Indicator = 0. Assertion ID 12279, Raw Assertion ID 351620, Sentence ID 1432008. The assertion relates concepts `goldfish` (ID 14183) and `carp` (ID 14184) with the relation `IsA` (ID 5). The sentence is `a goldfish is a carp.`. The score in each case is 16. This is the maximum score among all the cases in this class, and no other instance in this class achieves this score.

Indicator = 1. Assertion with ID 320114 has `best_raw_id` equal to null. The assertion relates the concept `drink` (ID 120) with itself with the relation `UsedFor` (ID 7). There are 832 assertions with null `best_raw_id`.

Assertion with ID 962 refers to raw assertion with ID 965, but there is no raw assertion with such an ID. The assertion relates the concepts `fight war` (ID 437) and `hate` (ID 1342) with the relation `HasPrerequisite` (ID 3), and has score equal to zero. There are 39312 different `best_raw_id`'s such that there is no raw assertion with such an ID. Apparently, all these IDs are mentioned only once in the table `conceptnet_assertion`.

Indicator = 2. Assertion ID 39773, Raw Assertion ID 42548, Sentence ID 787525. The assertion relates the concepts `baseball` (ID 1890) and `game` (ID 732) with the relation `IsA` (ID 5). The sentence is `Baseball is a game`. Assertion and raw assertion give a score of 16, while the sentence gives a score of 1. The half-discrepancy is 15. This is the maximum half-discrepancy that can be observed in this class, and no other triple can achieve this value.

Indicator = 3. Assertion ID 115013, Raw Assertion ID 127019, Sentence ID 942706. The assertion relates the concepts `jason` (ID 82025) and `late` (ID 1520) with the relation `HasProperty` (ID 20). The sentence is `jason is not late`. Assertion and raw assertion give a score of -2, while the sentence gives a score of 1. The half-discrepancy is 3. This is the maximum half-discrepancy that can be observed in this class. A similar half-discrepancy of 3 can be observed for the assertion with ID 544123.

Indicator = 4. Assertion ID 181807, Raw Assertion ID 72566, Sentence ID 840388. The assertion relates the concepts `jack` (ID 14299) and `child game` (ID 127337) with the relation `IsA` (ID 5). The sentence is `Jacks is a children's game that requires agility.`. Assertion and sentence give a score of 1, while the raw assertion gives a score of 10. The half-discrepancy is 9. This is the maximum half-discrepancy that can be observed in this class, and no other triple can achieve this value.

Indicator = 5. Assertion ID 197813, Raw Assertion ID 224508, Sentence ID 1177474. The assertion relates the concepts `marujuana` (ID 137113) and `cannabis` (ID 37883) with the relation `IsA` (ID 5). The sentence is `Marujuana is Cannabis`. Assertion and sentence give a score of 2, while the raw assertion gives a score of -2. The half-discrepancy is 4. This is the maximum half-discrepancy that can be observed in this class, and no other triple can achieve this value.

Indicator = 6. Assertion ID 56287, Raw Assertion ID 83533, Sentence ID 861172. The assertion relates the concepts `pen` (ID 1205) and `write` (ID 1893) with the relation `IsA` (ID 5). The sentence is `a pen is something you write with`. Raw assertion and sentence give a score of 1, while the assertion gives a score of 74. The half-discrepancy is 73. This is the maximum half-discrepancy that can be observed in this class, and no other triple can achieve this value.

Indicator = 7. Assertion ID 67530, Raw Assertion ID 176468, Sentence ID 1052796. The assertion relates the concepts `snake` (ID 369) and `leg` (ID 1252) with the relation `HasA` (ID 16). The sentence is `A snake does not have legs.`. Raw assertion and sentence give a score of 1, while the assertion gives a score of -7. The half-discrepancy is 8. This is the maximum half-discrepancy that can be observed in this class, and no other triple can achieve this value.

Indicator = 8. Assertion ID 1664, Raw Assertion ID 19218, Sentence ID 748040. The assertion relates the concepts `baseball` (ID 1890) and `sport` (ID 2130) with the relation `IsA` (ID 5). The sentence is `Baseball is a sport played in the U.S.`. Assertion gives a score of 147, raw assertion a score of 124, and sentence a score of 1. The half-discrepancy is 146. This is the maximum half-discrepancy that can be observed in this class, and no other triple can achieve this value.

Indicator = 9. Assertion ID 196090, Raw Assertion ID 222398, Sentence ID 1173220. The assertion relates the concepts `person` (ID 9) and `headache` (ID 2062) with the relation `Desires` (ID 10). The sentence is `a person wants a headache`. Assertion gives a score of 7, raw assertion a score of -2, and sentence a score of 13. The half-discrepancy is 15. This is the maximum half-discrepancy that can be observed in this class, and no other triple can achieve this value.

1.7 Third and Final Pass

In the third pass we parse the data in the tables conceptnet_concept and conceptnet_surfaceform. This allows us to load the concepts and the surface forms that were raised from the previous pass. In theory, it could be the

case that these new additional surface forms were referring to concepts that have not been raised yet from the previous passes, and hence we would require one more pass on the conceptnet_concept table to add these last concepts. However, this is not the case. In other words, these newly introduced surface forms from the last pass do not refer to concepts that we have not encountered earlier. Hence, this third pass is the last pass that we perform on the tables of the database.

Chapter 2

Consistency of the Database

The database is inconsistent. We have different assertions between the same concepts using the *same* relation but *different* frequency. Not only that, but the *value of the frequency can have opposite signs*, *implying* essentially *controversial statements*. Moreover, both statements can be characterized as correct since the score (measure of the validity of the statement) is positive in both cases!

Example 1. *We have the following instance.*

concept 1:	*man (id 7)*
concept 2:	*animal (id 902)*
relation:	*IsA (id 5)*

Assertion ID: *103395*

- *frequency: 1 (value: 5, string description: empty string)*
- *score: 3*
- *best raw assertion: 368795 (points to sentence 1672478)*
- *sentence: "man is a kind of animal."*

Assertion ID: *616165*

- *frequency: 25 (value: -5, string description: "not")*
- *score: 1*
- *best raw assertion: 827499 (points to sentence 2158613)*
- *sentence: "man is not animal"*

Part II

Structural Properties of ConceptNet 4

Chapter 3

High Level Overview and Conventions

In this part we will examine basic structural properties of `ConceptNet 4`. All the results are based on the `ConceptNet.db` file located in the `.conceptnet` directory under our home directory. Regarding the specifics of the .db file we have:

```
$ ls -l ~/.conceptnet/ConceptNet.db
-rw-r--r-- 1 user user 959354880 Feb 11  2010 /home/user/.conceptnet/ConceptNet.db
$ file ~/.conceptnet/ConceptNet.db
/home/user/.conceptnet/ConceptNet.db: SQLite 3.x database
$
```

3.1 Assertions

The `ConceptNet 4` database has $828,252$ assertions; $566,094$ are in English. These assertions define the input for the edges of the induced graphs.

Convention 2 (Input Definition)**.** *The input is defined by the assertions of the English language only.*

Remark 5. *The preliminary analysis will consider edges that have both negative and positive score. However, as the analysis progresses we will focus on edges that have strictly positive score, since the rest of the assertions have received at least one negative vote, and the number of negative votes is at least as the number of positive votes.*

3.2 Concepts

The `ConceptNet 4` database has $460,306$ concept IDs; $321,993$ are in English.

- The minimum concept ID found among the assertions of the English language is: 5 for `something`.
- The maximum concept ID found among the assertions of the English language is: $482,783$ for `understand human mind brain`.
- Number of different concepts appearing in assertions: $279,497$.
- Number of different concepts appearing in the closure of the input: $279,885$.
- Allowing self-loops there are $262,577$ different concepts with non-zero total degree on the induced subgraphs formed by edges with positive score.
- Disallowing self-loops there are $262,575$ different concepts with non-zero total degree on the induced subgraphs formed by edges with positive score.

Convention 3 (Number of Concepts in `ConceptNet 4`)**.** *In what follows, when we refer to the total number of concepts found in `ConceptNet 4`, we mean $279,497$, which is the number of concepts appearing in the assertions of the English language, which in turn define our input.*

3.3 Relations

`ConceptNet 4` has 30 relations; 27 appear among the assertions in the English language. Table A.9 in Appendix A.1 gives an overview of all the relations found in `ConceptNet 4`.

3.4 Frequencies

Table A.10 in Appendix A.1 presents the different frequencies that we can encounter in `ConceptNet 4` in the assertions of the English language.

3.5 Edges and Isolated Vertices in the Induced (Multi-)Graph Variants

Table 3.1 presents the number of edges as well as the isolated vertices that we encounter in 12 different cases in `ConceptNet 4`. The cases are 12 since we can distinguish cases based on the following:

- whether we allow edges with all scores or not,
- whether we allow self-loops or not,
- whether we allow edges with negative polarity, positive polarity, or finally both.

Table 3.1: Number of edges and isolated vertices on different variants of the induced subgraphs that can be obtained in `ConceptNet 4` by looking at the assertions of the English language. The marks ✓ and ✗ indicate respectively whether we allow self-loops in the induced (multi-)graphs or not. The enumeration allows all possible relations and frequencies on the edges.

score	self-loops	polarity	multigraph	directed graph	undirected graph	isolated vertices
all	✗	negative	15,327	15,168	14,707	267,187
all	✗	positive	550,277	465,866	452,445	5,764
all	✗	both	565,604	478,624	464,767	2
all	✓	negative	15,342	15,182	14,721	267,187
all	✓	positive	550,752	466,166	452,745	5,762
all	✓	both	566,094	478,929	465,072	0
positive	✗	negative	13,497	13,387	12,989	267,790
positive	✗	positive	478,499	412,956	401,367	22,651
positive	✗	both	491,996	424,525	412,569	16,922
positive	✓	negative	13,510	13,399	13,001	267,790
positive	✓	positive	478,879	413,216	401,627	22,649
positive	✓	both	492,389	424,790	412,834	16,920

3.6 Non-Zero Degrees and Self-Loops in the Induced (Multi-)Graph Variants

Again we distinguish four cases based on whether we include edges with all possible scores or not and on whether we allow self-loops or not. There are two nodes that have self-loops only among their edges. These are the nodes with IDs $56,959$ (`hansome` [1]) and $201,444$ (`needless death`). Table 3.2 gives an overview of the directed case, while Table 3.3 gives an overview of the undirected case. The entries for the number of vertices with non-zero degree in the undirected case are obtained by subtracting the number of isolated vertices found in Table 3.1 from $279,497$. Regarding the number of nodes that that have self-loops, these numbers are identical to the directed case which is presented in Table 3.2. However, we write down these numbers for clarity. Note that the numbers found in these column refer to vertices and are not counting distinct self-loops. Counting distinct self-loops in different cases will be examined in Section 3.7.

[1] This is the actual spelling of the concept.

Table 3.2: Overview on the degrees of the induced directed multigraphs and graphs for `ConceptNet 4` in the English language.

			number of nodes with				
score	self-loops	polarity	$\neq 0$ in-deg	$\neq 0$ out-deg	$\neq 0$ in-/out-deg	self-loops	self-loops only
all	✗	negative	$9,291$	$4,412$	$1,393$	–	–
all	✗	positive	$233,456$	$60,628$	$20,351$	–	–
all	✗	both	$238,389$	$61,839$	$20,733$	–	–
all	✓	negative	$9,293$	$4,412$	$1,395$	14	0
all	✓	positive	$233,462$	$60,634$	$20,361$	300	2
all	✓	both	$238,395$	$61,845$	$20,743$	305	2
positive	✗	negative	$8,884$	$4,041$	$1,218$	–	–
positive	✗	positive	$216,198$	$60,052$	$19,404$	–	–
positive	✗	both	$221,114$	$61,241$	$19,780$	–	–
positive	✓	negative	$8,886$	$4,041$	$1,220$	12	0
positive	✓	positive	$216,204$	$60,057$	$19,413$	260	2
positive	✓	both	$221,120$	$61,246$	$19,789$	265	2

3.7 Decomposition of Assertions and Edges

Table 3.4 gives the decomposition of the assertions in the English language.

3.7.1 Partitioning Edges with Positive Score with respect to Frequencies

Here we examine the number of edges of the induced subgraphs according to different frequency value ranges. In every case we retain the edges with strictly positive score. According to Convention 3 the number of nodes is $279,497$ in every case. Moreover, note that the number of edges of the induced multigraph with frequency values in the range $\{-10,\ldots,0\}$ plus the number of edges of the induced multigraph with frequency values in the range $\{0,\ldots,10\}$ is equal to $13,510+478,879 = 492,389$ which agrees with the total number of edges with positive score mentioned in Table 3.1.

Table 3.5 gives a detailed overview in every case. Note that from Table 3.5 it follows that there are no edges with values for frequencies from the set $\{-9,-8,-7,-6,-4,-3,-1,0,1,3,6\}$, which is, as it should be, in complete agreement with Table A.10.

Table 3.3: Overview of the degrees for the induced undirected multigraphs and graphs for `ConceptNet 4` in the English language. The columns about self-loops refer to the multigraph only.

			number of nodes with		
score	self-loops	polarity	$\neq 0$ degree	self-loops	self-loops only
all	✗	negative	12,310	–	–
all	✗	positive	273,733	–	–
all	✗	both	279,495	–	–
all	✓	negative	12,310	14	0
all	✓	positive	273,735	300	2
all	✓	both	279,497	305	2
positive	✗	negative	11,707	–	–
positive	✗	positive	256,846	–	–
positive	✗	both	262,575	–	–
positive	✓	negative	11,707	12	0
positive	✓	positive	256,848	260	2
positive	✓	both	262,577	265	2

Table 3.4: Decomposition of assertions in the English language found in `ConceptNet 4`. We consider all assertions regardless of their score and all assertions with positive score. Next to the number of edges or self-loops for each relation we see, in that order, how many have negative polarity and how many have positive polarity.

			induced directed multigraph based on assertions with			
	relation		all scores		positive score	
index	id	name	edges	self-loops	edges	self-loops
0	1	HasFirstSubevent	4192 (4/4188)	1 (0/1)	4121 (3/4118)	1 (0/1)
1	2	HasLastSubevent	3066 (8/3058)	2 (0/2)	2971 (8/2963)	2 (0/2)
2	3	HasPrerequisite	23801 (68/23733)	56 (0/56)	23404 (55/23349)	56 (0/56)
3	4	MadeOf	1662 (29/1633)	5 (1/4)	1545 (25/1520)	4 (1/3)
4	5	IsA	111547 (4797/106750)	89 (11/78)	94726 (3884/90842)	73 (10/63)
5	6	AtLocation	49508 (973/48535)	43 (0/43)	45192 (764/44428)	26 (0/26)
6	7	UsedFor	52135 (276/51859)	31 (1/30)	50451 (194/50257)	30 (1/29)
7	8	CapableOf	40141 (2994/37147)	13 (0/13)	39391 (2924/36467)	11 (0/11)
8	9	MotivatedByGoal	15312 (33/15279)	36 (0/36)	15116 (27/15089)	35 (0/35)
9	10	Desires	9295 (4083/5212)	4 (1/3)	9059 (4048/5011)	3 (1/2)
10	12	ConceptuallyRelatedTo	23097 (0/23097)	21 (0/21)	23010 (0/23010)	21 (0/21)
11	13	DefinedAs	6500 (3/6497)	7 (0/7)	6428 (0/6428)	7 (0/7)
12	14	InstanceOf	70 (0/70)	0 (0/0)	69 (0/69)	0 (0/0)
13	15	SymbolOf	167 (0/167)	0 (0/0)	166 (0/166)	0 (0/0)
14	16	HasA	55311 (415/54896)	41 (0/41)	22786 (399/22387)	11 (0/11)
15	17	CausesDesire	5179 (20/5159)	3 (0/3)	4989 (15/4974)	2 (0/2)
16	18	Causes	18624 (53/18571)	25 (0/25)	18257 (34/18223)	24 (0/24)
17	19	HasSubevent	26206 (119/26087)	19 (0/19)	25444 (93/25351)	18 (0/18)
18	20	HasProperty	93384 (1447/91937)	63 (0/63)	82458 (1027/81431)	53 (0/53)
19	21	PartOf	4935 (13/4922)	13 (0/13)	4676 (9/4667)	8 (0/8)
20	22	ReceivesAction	10907 (1/10906)	5 (1/4)	10848 (0/10848)	3 (0/3)
21	24	InheritsFrom	185 (0/185)	2 (0/2)	64 (0/64)	2 (0/2)
22	25	CreatedBy	586 (6/580)	2 (0/2)	557 (1/556)	1 (0/1)
23	28	HasPainCharacter	34 (0/34)	0 (0/0)	34 (0/34)	0 (0/0)
24	29	HasPainIntensity	74 (0/74)	0 (0/0)	73 (0/73)	0 (0/0)
25	30	LocatedNear	5053 (0/5053)	1 (0/1)	5044 (0/5044)	1 (0/1)
26	31	SimilarSize	5123 (0/5123)	8 (0/8)	1510 (0/1510)	1 (0/1)
		total	566094 (15342/550752)	490 (15/475)	492389 (13510/478879)	393 (13/380)

Table 3.5: Number of edges in the induced subgraphs for various frequency ranges. The columns in the cases of multigraph, directed graph, and undirected graph present the number of edges with and without self-loops (in that order) in every case. All relations are allowed between the concepts but the scores of the assertions have to be positive.

polarity	range for frequency values	number of edges with and without self-loops					
		multigraph		directed graph		undirected graph	
negative	{-10}	187	187	187	187	187	187
	{-10, -9}	187	187	187	187	187	187
	{-10, -9, -8}	187	187	187	187	187	187
	{-10, ..., -7}	187	187	187	187	187	187
	{-10, ..., -6}	187	187	187	187	187	187
	{-10, ..., -5}	13,395	13,382	13,287	13,275	12,889	12,877
	{-10, ..., -4}	13,395	13,382	13,287	13,275	12,889	12,877
	{-10, ..., -3}	13,395	13,382	13,287	13,275	12,889	12,877
	{-10, ..., -2}	13,510	13,497	13,399	13,387	13,001	12,989
	{-10, ..., -1}	13,510	13,497	13,399	13,387	13,001	12,989
	{-10, ..., 0}	13,510	13,497	13,399	13,387	13,001	12,989
positive	{0, ..., 10}	478,879	478,499	413,216	412,956	401,627	401,367
	{1, ..., 10}	478,879	478,499	413,216	412,956	401,627	401,367
	{2, ..., 10}	478,879	478,499	413,216	412,956	401,627	401,367
	{3, ..., 10}	478,872	478,492	413,209	412,949	401,620	401,360
	{4, ..., 10}	478,872	478,492	413,209	412,949	401,620	401,360
	{5, ..., 10}	471,543	471,170	407,244	406,987	395,726	395,469
	{6, ..., 10}	4,930	4,930	4,860	4,860	4,859	4,859
	{7, ..., 10}	4,930	4,930	4,860	4,860	4,859	4,859
	{8, 9, 10}	2,217	2,217	2,206	2,206	2,205	2,205
	{9, 10}	445	445	444	444	443	443
	{10}	386	386	385	385	384	384

Chapter 4

Degrees and Distributions

Here we examine the degrees and the degree distributions on the various induced graphs. Figure 4.1 gives a snapshot of the total degree distribution of the induced directed multigraph of `ConceptNet 4` as this was generated by Wordle [1]. Table 4.1 presents the 100 concepts with highest total degree in the same graph (that is, directed multigraph induced by the assertions of the English language with positive score).

Figure 4.1: The 2013 concepts with highest total degree in the directed multigraph induced by assertions of positive score in the English language. For clarity the total degree for the concept `person` was scaled down to 1/3 of its actual value (see Table 4.1) for better visual output.

[1] Homepage: `http://www.wordle.net`

Table 4.1: The 100 concepts with the highest total degree in the directed multigraph induced by the assertions with positive score in the English language.

concept	degree	concept	degree	concept	degree	concept	degree	concept	degree
person	19,172	food	1,042	metal	784	walk	661	game	588
something	2,893	cat	1,010	read	781	wind	655	doctor	583
human	1,794	exercise	986	cake	760	birthday	648	die	581
this	1,637	animal	971	rest	756	kill	643	bar	578
child	1,500	eat	960	sleep	752	garden	643	oil	573
fun	1,378	drink	927	talk	750	build	642	store	568
water	1,366	home	906	bed	749	apple	638	room	567
book	1,241	fish	881	bird	744	examine	638	sound	564
it	1,208	computer	876	smoke	732	record	633	swim	563
man	1,204	paper	865	wood	732	cook	625	card	562
dog	1,152	plant	846	school	730	table	620	baby	561
money	1,133	city	832	time	714	verb	620	drive car	558
party	1,128	plate	825	country	708	boat	618	finger	544
paint	1,124	play	818	chicken	704	fire	615	live	541
music	1,123	work	807	squirrel	700	flower	615	love	541
horse	1,122	tree	801	glass	695	door	610	surprise	540
car	1,114	eye	798	buy	689	body	610	machine	540
write	1,095	drive	796	woman	684	run	604	shade	537
house	1,089	learn	793	hand	683	desk	595	corn	529
dance	1,076	farm	793	think	672	sex	589	earth	528

4.1 Average Degrees

The average degree in every case is given by $2|E|/|V|$. Regarding the number of edges we use the entries found in Table 3.1. As of the number of vertices, we use both 279,497 which is the amount of concepts appearing among all the assertions in the English language regardless of the score of the assertions (Convention 3), as well as the smaller values that are obtained when we subtract from that number the number of the isolated vertices that is given in Table 3.1. The multigraph has an average degree of roughly 3.6, the directed graph of roughly 3.1, and the undirected graph of roughly 3.0. Table 4.2 gives the details in every case.

4.2 Degree Distribution

Figure 4.2 gives the degree distribution for the directed multigraph induced by assertions with positive score in three cases. Recall that the polarity of the assertions can be both positive and negative. Hence the three cases that we distinguish in the plots in Figure 4.2 are for the cases where:

- arbitrary polarity is allowed; that is both positive and negative,
- negative only polarity is allowed, and
- positive only polarity is allowed.

The initial segment of the total-degree distribution (edges with both positive and negative polarity are allowed) is given in Table 4.3.

4.2.1 Fitting

We investigate the three networks presented in Figure 4.2 using the *method of maximum likelihood* suggested in [3] and the relevant tools [2] that are available online [3]. The script that we use for power law fitting, is the

[2] http://tuvalu.santafe.edu/~aaronc/powerlaws/

[3] We urge the reader to go through http://vserver1.cscs.lsa.umich.edu/~crshalizi/weblog/491.html as well.

Table 4.2: The average degree of the induced multigraphs and graphs of `ConceptNet 4`. All values are rounded in the third decimal point. The number of vertices in the induced graphs is considered to be equal to $279,497$. Inside the parentheses we see the values that are obtained when we subtract from those vertices the number of isolated vertices as these are described in Table 3.1.

score	self-loops	polarity	directed multigraph	directed graph	undirected graph
all	✗	negative	0.110 (2.490)	0.109 (2.464)	0.105 (2.389)
all	✗	positive	3.938 (4.021)	3.334 (3.404)	3.238 (3.306)
all	✗	both	4.047 (4.047)	3.425 (3.425)	3.326 (3.326)
all	✓	negative	0.110 (2.493)	0.109 (2.467)	0.105 (2.392)
all	✓	positive	3.941 (4.024)	3.336 (3.406)	3.240 (3.308)
all	✓	both	4.051	3.427	3.328
positive	✗	negative	0.097 (2.306)	0.096 (2.287)	0.093 (2.219)
positive	✗	positive	3.424 (3.726)	2.955 (3.216)	2.872 (3.125)
positive	✗	both	3.521 (3.747)	3.038 (3.234)	2.952 (3.142)
positive	✓	negative	0.097 (2.308)	0.096 (2.289)	0.093 (2.221)
positive	✓	positive	3.427 (3.729)	2.957 (3.218)	2.874 (3.127)
positive	✓	both	3.523 (3.750)	3.040 (3.236)	2.954 (3.144)

Table 4.3: The initial segment of the total-degree distribution in the directed multigraph induced by the assertions of the English language with positive score. The values shown for the frequencies in the third row are merely the numerical values obtained from the quotient $\frac{\text{number of concepts with degree } d}{279{,}497}$, where $279,497$ is the number of nodes for the entire network according to Convention 3.

degree	0	1	2	3	4	5	6	7	8	9	...
concepts	16,920	203,556	26,775	9,880	4,959	2,968	1,962	1,415	1,007	802	...
frequency	0.060537	0.728294	0.095797	0.035349	0.017743	0.010619	0.007020	0.0050627	0.003603	0.002869	...

implementation of Tamás Nepusz [4], version 0.7. A typical execution of the script for the results presented below.

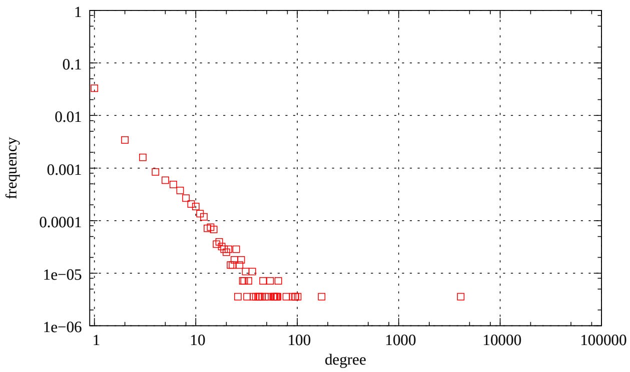
```
$ plfit -M -p approximate inputFile
```

Hence we also get the first four central moments of the degree distribution, as well as calculate an approximate p-value. Note that in order for the input to make sense all the concepts that are part of the input should have degree at least 1. In other words, we have to omit from the input all the isolated vertices. Detailed results for every case are presented in Table 4.4. Using `plplot` by Joel Ornstein we obtain the figures shown in Figure 4.4.

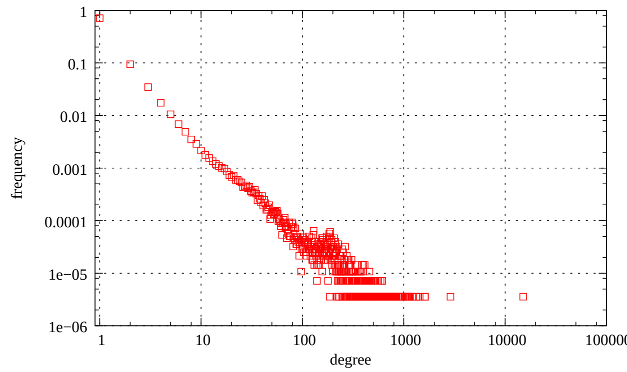
Table 4.4: Fitting power law in the degree distributions of `ConceptNet 4` on the multigraphs induced by the assertions with negative polarity only, positive polarity only, or both. The exponent (scaling) is denoted by α, x_{min} is the lower bound to the power law behavior, $\mathcal{L}$ is the maximum log-likelihood, D is the Kolmogorov-Smirnov (or KS) statistic, and p is for the p-value.

polarity	α	x_{min}	$\mathcal{L}$	D	p	mean	variance	std. dev.	skewness	kurtosis
negative	2.77868	10	−994.91	0.01532	0.0082	2.308	1,450.692	38.088	106.245	11,423.245
positive	1.82643	5	−66,869.11	0.02699	0.0000	3.729	1,488.787	38.585	239.212	90,850.012
both	1.82572	5	−68,098.45	0.02646	0.0000	3.750	2,021.043	44.956	300.041	126,012.584

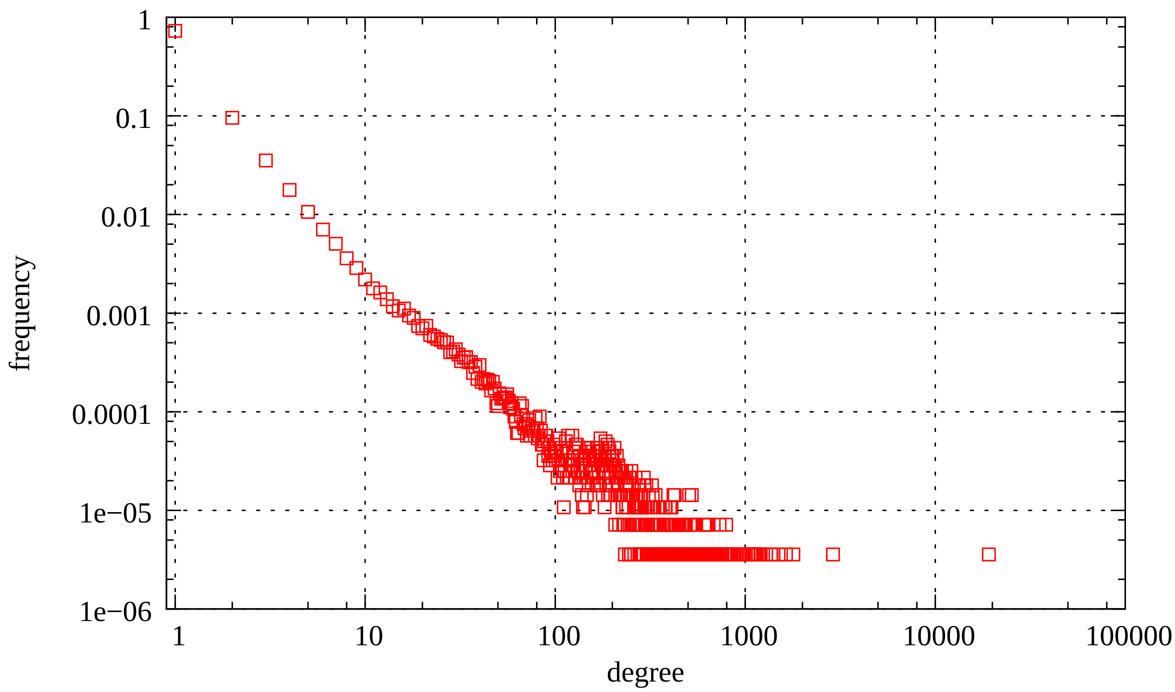
[4] https://github.com/ntamas/plfit



(a) Degree distribution when only negative polarity is taken into account.

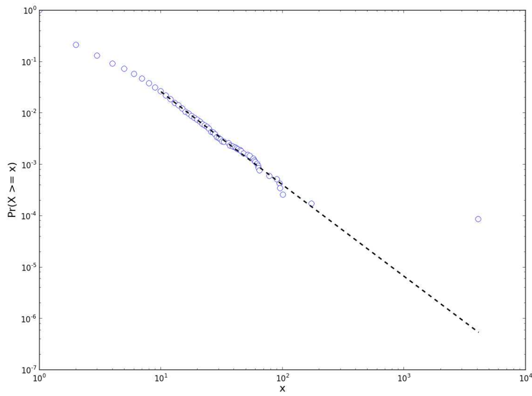


(b) Degree distribution when only positive polarity is taken into account.

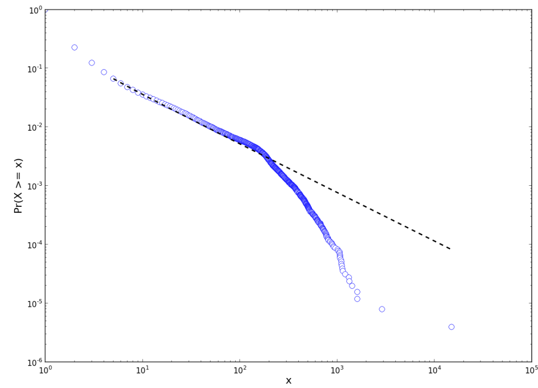


(c) Degree distribution when all polarities are taken into account.

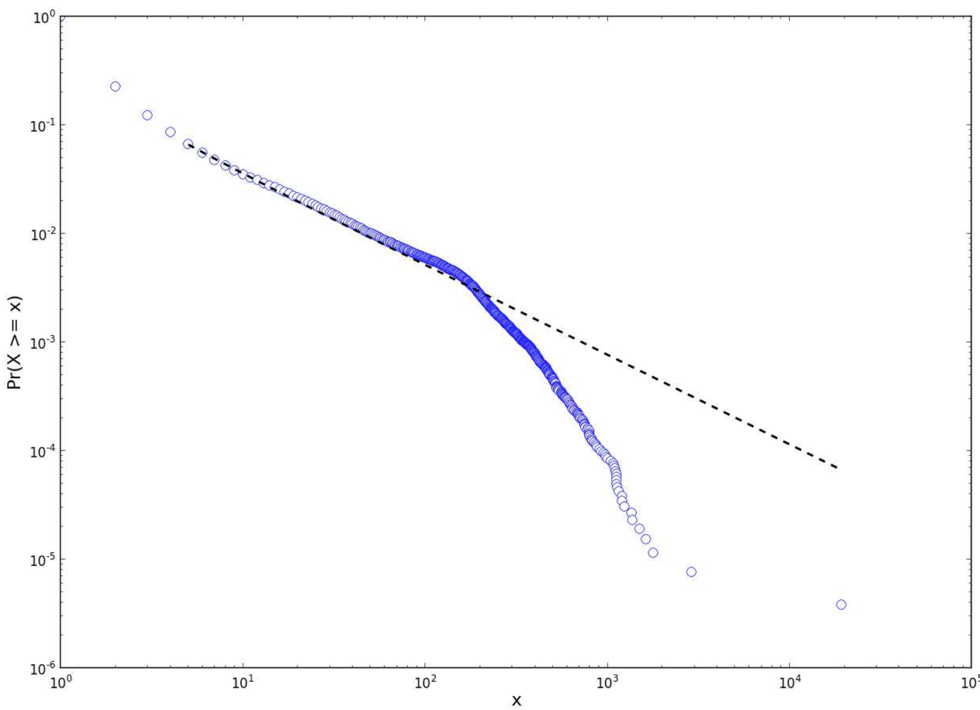
Figure 4.2: Degree distributions in three different cases for the induced directed multigraph. In every case we take into account only the assertions of the English language with positive score. The different cases arise if we further want to differentiate and take into account assertions with negative polarity only, positive polarity only, or finally arbitrary polarity.



(a) Power law fitting when only negative polarity is taken into account.



(b) Power law fitting when only positive polarity is taken into account.



(c) Power law fitting when all polarities are taken into account.

Figure 4.3

Figure 4.4: Power law fitting in the three major degree distributions of `ConceptNet 4` using the method of maximum likelihood presented in [3].

Chapter 5

Connected Components, Transitivity, and Clustering Coefficient

For the computations found in this chapter we are going to neglect self-loops in the directed or undirected graphs induced by the assertions of the English language with positive score. The reason is that self-loops do not affect the connectivity of the components. We use the function `igraph_clusters` of `igraph` [4] to compute the connected components of the graphs.

Definition 3 (Global Transitivity [15])**.** *Transitivity measures the probability that two neighbors of a vertex are connected. More precisely, it is the ratio of the triangles and connected triples in the graph.*

Definition 4 (Average Local Transitivity or Clustering Coefficient [16])**.** *The average local transitivity also measures the probability that two neighbors of a vertex are connected. However, in case of the average local transitivity, this probability is calculated for each vertex and then the average is taken. Vertices with less than two neighbors require special treatment; they will either be left out from the calculation, or they will be considered as having zero transitivity. Note that this measure is different from the global transitivity measure mentioned above as it simply takes the average local transitivity across the whole network. See [16] for more details.*

Clustering coefficient is an alternative name for transitivity [4]. In this document we will imply the average local transitivity whenever we refer to the clustering coefficient.

5.1 Transitivity and Clustering Coefficient

Table 5.1 presents the transitivity and the clustering coefficient for the undirected graph induced by the assertions of the English language with positive score neglecting self-loops.

Table 5.1: Transitivity and clustering coefficient for the entire graph of `ConceptNet 4` induced by assertions with negative only polarity, positive only polarity, and both polarities. The first value (NAN) for the clustering coefficient gives the result of the calculation when vertices with less than two neighbors are left out from the calculation, while the second value (ZERO) gives the result of the calculation when vertices with less than two neighbors are considered as having zero transitivity. Note that all values are the same both for directed as well as undirected graphs.

	polarity		
	negative	positive	both
Transitivity	0.000351298700593188	0.004964054809387655	0.003881697564836174
Clustering Coefficient (NAN)	0.098300551193575281	0.190154012754549323	0.196101493828584605
Clustering Coefficient (ZERO)	0.000851478314399245	0.032692554034191273	0.034448280818741697

5.2 Negative Polarity: Connected Components

First we examine the case of the directed and undirected graph induced by the assertions with negative polarity.

5.2.1 Weakly Connected Components

We get $269,167$ weakly connected components, out of which $267,790$ are isolated vertices. Note that $267,790$ is in complete agreement with Table 3.1. Among the rest $1,377$ components we can find components with cardinalities between 2 and $8,596$.

Distribution of Component Sizes. The distribution of the sizes for the various components is shown in Table 5.2. This distribution presents the cardinalities of the weakly connected components of the induced directed graph, as well as the cardinalities of the connected components of the induced undirected graph.

Table 5.2: Distribution of sizes for weakly connected components for the induced directed graph. This is also the distribution of sizes for the connected components of the induced undirected graph.

# of nodes	8,596	13	9	8	7	6	5	4	3	2	1
# of components	1	2	2	2	4	8	22	28	137	1,171	267,790

Figure 5.1 presents the maximal weakly connected component of size $8,596$. Figure 5.2 presents the weakly connected components with sizes 8, 9 and 13.

Big Weakly Connected Component

The undirected graph induced by the concepts that appear in the big undirected component is composed of $8,596$ nodes and $11,247$ edges. For information about shortest paths in this component please see Chapter 7.

Components of Size 13

In the first component of size 13 concept `may` (2606) has out-degree 6 and in-degree 1. Concepts `April` (2721), `make right` (2766), and `weak` (21769) have out-degree 0 and in-degree 2. Concepts `march` (2719) and `will` (20015) have out-degree 2 and in-degree 0. Concepts `February` (2716), `June` (2725), `definite` (37022), `know definition word 'hemisphere` (328106), and `wont` (333527) have out-degree 0 and in-degree 1. Concepts `two wrong` (2765) and `feminine woman` (109816) have out-degree 1 and in-degree 0.

In the second component of size 13 concept `division` (14946) has out-degree 7 and in-degree 1. Concepts `union` (4832) and `add` (54627) have out-degree 3 and in-degree 1. Concepts `addition` (26573) and `subtract` (108338) have out-degree 0 and in-degree 3. Concept `subtraction` (161354) has out-degree 1 and in-degree 2. Concept `multiplication` (14387) has out-degree 2 and in-degree 1 Concept `multiply` (25479) has out-degree 0 and in-degree 2. Concept `divide` (19901) has out-degree 2 and in-degree 0. The rest four concepts `intersection` (5593), `minus` (332948), `confederacy` (351157), and `confederate` (369189) all have out-degree 0 and in-degree 1.

Components of Size 9

In the first component of size 9 concept `if person` (48339) has out-degree 8 and in-degree 0. All the other concepts have out-degree 0 and in-degree 1. These 8 concepts are `water plant die` (20613), `green card i.n.` (58241), `read never succeed` (74287), `two telephon` (120923), `pay bill go bankrupt` (128794), `go out stay home` (189195), `beat join` (201293), and `license not drive` (428525).

In the second component of size 9 concept `topic 'sky` (311764) has out-degree 7 and in-degree 0. Concept `word drop` (20977) has out-degree 0 and in-degree 2. Concepts `word metal-frame` (21208), `word aurora` (21870), `word pressure` (22718), `word sagittarius` (22726), `word high` (23450), and `word helium` (23546) have out-degree 0 and in-degree 1. Finally, concept `topic 'liquid` (311723) has out-degree 1 and in-degree 0.

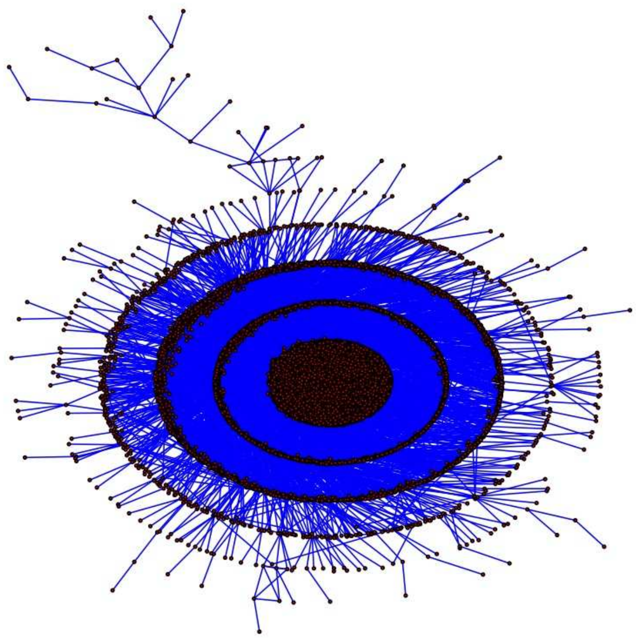


Figure 5.1: The maximal weakly connected component of the graph induced by the assertions with negative polarity; see Table 5.2. For simplicity we plot the induced undirected graph of that component.

Components of Size 8

In the first component of size 8 concept `comfortable` (371) has out-degree 0 and in-degree 5. Concepts `classroom chair` (26777) and `wicker` (34790) have out-degree 2 and in-degree 0. Concepts `sturdy oak` (51878) and `build comfort` (175882) have out-degree 0 and in-degree 1. Finally the concepts `sofabed` (4007), `chair make outdoor use` (59113), and `sleep couch` (138393) have out-degree 1 and in-degree 0.

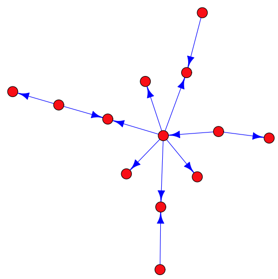
In the second component of size 8 concept `good eat` (2543) has out-degree 0 and in-degree 7. All the other concepts have out-degree 1 and in-degree 0. These 7 concepts are `hair gel` (3104), `yellow snow` (24319), `cosmetic` (47806), `orange peel` (63084), `crabapple` (103589), `unripe orange` (117785), and `peel orange` (117790).

5.2.2 Strongly Connected Components

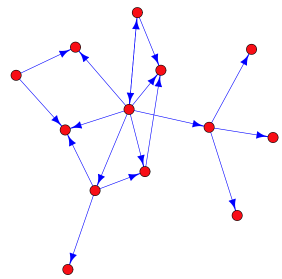
We get 278,783 strongly connected components, out of which 278,708 are isolated vertices. Among the rest 75 components we can find components with cardinalities between 2 and 592.

Distribution of Component Sizes. The distribution of the sizes for the various components is shown in Table 5.3. This distribution presents the cardinalities of the strongly connected components of the induced directed graph.

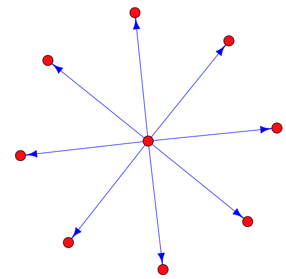
Figure 5.3 presents the maximal strongly connected component.



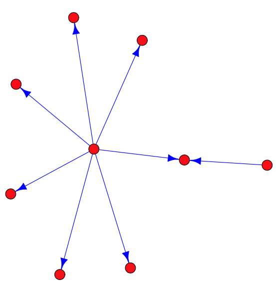
(a) may, April, make right, weak, march, will
13 nodes, 12 edges.



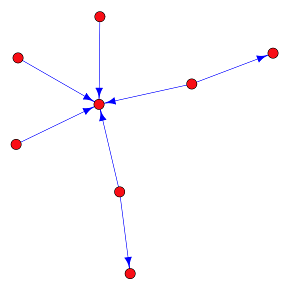
(b) division, union, add, addition, subtract, subtraction, multiplication, multiply, divide
13 nodes, 18 edges.



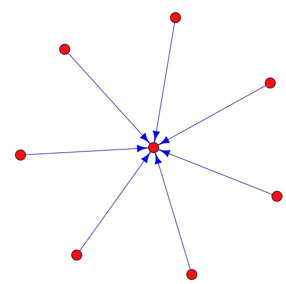
(c) if person
9 nodes, 8 edges.



(d) topic 'sky, word drop
9 nodes, 8 edges.



(e) comfortable, classroom chair, wicker
8 nodes, 7 edges.



(f) good eat
8 nodes, 7 edges.

Figure 5.2: The induced *directed* subgraphs for some weakly connected components; see Table 5.2. The names of the subgraphs are given by the nodes with total degrees different from 1. All such nodes are listed in decreasing order of total degree. In case of ties precedence takes the name of the node that has larger in-degree.

Big Strongly Connected Component

The 592 concepts found in the big directed component are man (7), person (9), rock (23), beach (24), tree (33), work (35), actor (47), exercise (61), pant (63), love (67), library (68), bath (70), listen (75), wife (76), arm (79), human (80), run marathon (101), drink (120), examination (121), fun (134), it (137), paper (149), destroy (150), bed (156), dirty (170), dream (172), shower (173), child (178), smoke (188), chicken (191), blind (233), ball (263), mother (301), party (307), rest (310), remember (325), forget (326), housework (343), clean (344), street (350), watch tv (351), park (365), trouble (366), wood (370), play (372), bus (377), talk (394), go bed (406), sleep (425), eat (432), nothing (466), computer (467), rich (469), lover (472), buy (475), hunger (478), milk (481), sometimes (526), car (529), dog (537), music (542), film (544), zoo (547), dress (562), checkbook holder (563), bottle (565), better (570), live (580), one (581), aluminum (590), chair (596), skirt (601), drug (610), cat (616), gun (635), country (640), sell (649), house (652), fish (655), lake (660), baby (678), beauty (702), plant (716), silence (730), hide (869), girl (876), muscle (891), woman (895), animal (902), family (915), moon (924), robot (935), bird (962), death (977), play sport (983), sick (991), drive car (1005), bathroom (1007), city (1013), water (1016), truck (1028), desk (1043), office (1044), home (1045), bat (1057), penny (1071), couch (1072), build (1104), spoon (1116), travel (1143), eye (1160), see (1161), fail

Table 5.3: Distribution of sizes for strongly connected components for the induced directed graph.

# of nodes	592	12	10	8	5	4	3	2	1
# of components	1	1	2	1	1	4	6	59	278,708

(1167), nose (1171), smell (1172), well (1201), pen (1205), mine (1210), die (1227), money (1240), bill (1245), snow (1247), leg (1252), triangle (1257), dead (1279), lady (1281), mouse (1284), cry (1291), television (1298), hate (1342), sea (1347), ocean (1349), sun (1353), sky (1354), food (1359), lie (1395), horse (1412), mug (1422), friend (1429), grocery store (1447), read (1456), hold (1464), kill (1466), break (1476), guy (1479), foot (1485), newspaper (1506), late (1520), hungry (1533), drive (1545), liquid (1551), oil (1587), plate (1604), smile (1606), cow (1613), earth (1633), dance (1667), potato (1674), fight (1675), curtain (1694), glass (1776), telephone (1790), pain (1813), audience (1816), soul (1835), drop (1846), bone (1852), meat (1853), rain (1856), body (1861), write (1893), pencil (1953), book (2033), black (2063), fart (2079), honest (2087), profit (2167), complete (2201), close (2222), bad (2226), heaven (2241), show (2243), trash (2260), can (2261), gold (2266), wind (2284), hand (2300), debt (2306), stop (2358), road (2368), brother (2383), boat (2389), lose (2426), war (2438), flower (2459), wallet (2466), suitcase (2479), time (2494), problem (2500), hell (2510), small (2536), bicycle (2554), need (2557), enemy (2558), continent (2580), iron (2587), cookie (2595), color (2611), white (2612), red (2614), yellow (2616), colour (2626), stone (2631), vegetable (2636), green (2637), life (2638), murder (2663), wrong (2664), good (2666), evil (2692), large (2771), shoe (2790), go (2801), sex (2825), wait (2858), steak (2878), fire (2895), exist (2907), government (2932), beer (3052), none (3387), carpet (3450), bowl (3463), freedom (3492), born (3501), leave (3571), coin (3573), fruit (3590), laugh (3635), sister (3656), laundry (3665), fork (3671), planet (3683), shirt (3686), begin (3695), steel (3907), sidewalk (3962), avenue (4000), theatre (4095), cup (4116), square (4138), busy (4163), full (4189), pleasure (4231), god (4277), care (4323), star (4324), watch (4406), mind (4432), space (4435), wealth (4521), this (4539), place (4570), apple (4596), pear (4624), win (4676), mail (4691), direct (4753), doctor (4760), theater (4770), river (4784), blue (4808), charge (4811), cheese (4844), whale (4849), mammal (4850), question (4898), crap (4899), lot (4905), coal (5090), touch (5106), noise (5363), husband (5415), plastic (5505), bug (5563), above (5572), unknown (5613), matter (5619), disease (5645), table (5665), peace (5670), key case (5678), often (5700), sing (5711), sand (5768), billfold (5827), bottom (5887), religion (5915), long hair (5916), closet (5967), boy (5976), like (5989), record (6029), find (6040), floor (6062), right (6079), old (6092), safety (6244), cut (6250), honesty (6288), slow (6291), frustrate (6309), adult (6329), conflict (6331), here (6352), bite (6368), science (6395), air (6408), lime (6416), banana (6422), metal (6491), do (6503), open (6539), quiet (6583), big (6604), present (6681), roll (6734), mess (6818), mineral (6835), clock (6860), black hole (6876), chaos (6892), distance (6929), many (6989), competitive activity (7019), safe (7045), still (7048), violence (7055), round (7057), computer language (7112), mercury (7120), art (7424), rust (7512), top (7514), wine (7522), jar (7524), crowd (7763), draw (7764), much (7917), thing (7936), energy (7982), land (8060), few (8145), musician (8244), little (8268), change (8313), ear (8314), bread (8404), dna (8405), pick (8494), gasoline (8502), petrol (8691), move (8737), try (8794), decide (8824), tin (8891), finish (8996), fear (9006), poverty (9116), island (9131), shade (9151), fly (9215), hear (9269), egg (9339), penis (9458), vagina (9464), two (9549), dad (9672), health (9745), pass (9934), wash (10170), sock (10193), head (10228), work hard (10313), his (10419), fill (10468), great (10478), end (10507), know (13183), program language (13345), daughter (13446), supermarket (13550), danger (13607), servant (13683), silver (13722), pie (13747), machine (13790), gas (13908), taste (14093), ant (14190), fix (14209), lemon (14212), gerbil (14223), dollar (14251), want (14319), galaxy (14379), circle (14472), cake (14522), blood (14713), law (14805), copy (14847), chick (14872), illusion (14991), cent (14994), orange (15004), large bird (15149), dirt (15359), son (15379), fast (15507), point (15518), choose (15533), fact (15578), be (16974), software (17383), common (17473), brain (17555), liar (17830), stay (18183), sickness (18244), performance (18289), motion (18365), whole (18374), opinion (18525), out (18546), in (18553), return (18569), ill (18575), victory (18635), fantasy (18637), propose woman (18678), ignorance (18746), ride (18753), free (19126), past (19235), new (19512), part (19708), course (19871), rat (19911), own (19972), dog die (20317), vegetarian (20339), owner (20525), over (20622), real (20645), same (20650), best (20709), necessary (20908), box office (20927), poor (20993), car key (21233), angel (21240), stage (21403), order (21418), print (21683), microsoft (21796), rush (21894), empty (22345), freeway (22365), go break (22388), artifact

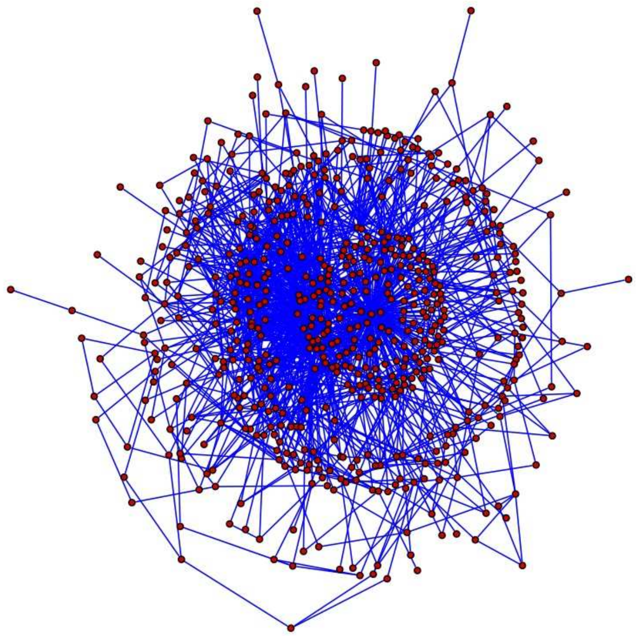


Figure 5.3: The maximal strongly connected component of the graph induced by the assertions with negative polarity; see Table 5.3. For simplicity we plot the induced undirected graph of that component.

(22487), `chore` (22621), `clear` (22671), `trip` (22700), `all` (22948), `cash register` (23016), `worse` (23274), `deaf` (23417), `truth` (23426), `conscious` (23506), `compassion` (23996), `food can` (24253), `elevator` (24427), `reality` (24722), `future` (24821), `loss` (24845), `orchestra pit` (24852), `sunshine` (25192), `answer` (25710), `solution` (25749), `rend` (26018), `master` (26090), `lord` (26283), `gentleman` (26487), `below` (27409), `slave` (27415), `brass` (27632), `come` (28590), `imaginary` (28877), `urban` (29003), `nurse` (29051), `away` (29340), `flat tire` (29840), `pest` (29938), `reply` (30251), `neglect` (30447), `lift` (30476), `run treadmill` (30496), `paste` (30733), `inch` (31249), `seek` (31416), `ask` (31437), `urine` (31765), `hatred` (32372), `itch` (33681), `some` (34413), `half` (34484), `checkbook cover` (34526), `ally` (34528), `bob` (34599), `bronze` (34633), `defeat` (34651), `computer virus` (34745), `enter` (36183), `shout` (36617), `park bench` (37143), `fine` (37229), `far` (37745), `miss` (38484), `vague` (38678), `balcony seat` (38825), `youth` (39134), `table cloth` (39246), `lint` (39879), `continue` (43163), `always` (43553), `early` (44789), `start` (44963), `kleenex` (47422), `okay` (47521), `now` (47894), `movie screen` (48200), `real duck` (48788), `monkey wrench` (51349), `recent` (52116), `ancient` (53725), `fondue` (55652), `snub` (56060), `myth` (57756), `agent` (58122), `enough` (60838), `apathy` (65195), `frown` (66280), `sandal` (70731), `bowie` (71734), `insure` (73501), `gain` (73685), `plasma` (75809), `any` (76623), `zero` (79459), `beast` (79780), `cost` (81860), `arrive` (88674), `gray` (93729), `celibate` (96998), `integer` (100946), `graze` (102383), `lease` (102411), `never` (126958), `modern` (131226), `idle` (138412), `illiteracy` (140633), `high octane` (144079), `occasional` (155305), `mistress` (174612), `entire` (177172), `ground` (184976), `rural` (185019), `under` (193674), `transportation device` (200905), `barack obama` (201863), `speedo` (203600), `fidelity` (203658), `norway rat` (203664), `complete thesis` (203671), `pour hot coffee mug` (203692), `low blood pressure` (203696), `cost little money` (203700), `broken` (311852), `written programmer` (328244), `foe` (332239), `mister` (332244), and `obscure` (344371).

Regarding the big strongly connected component with the 592 nodes, it has 1,849 edges (self-loops were omitted from the enumeration). Hence the average degree is about 6.24662 after self-loops have been discarded.

Regarding the induced undirected graph that occurs after restricting ourselves in these 592 nodes (again, self-loops are omitted), the number of edges is 1,566. In other words, the average degree in this case is about 5.29054. The transitivity and the clustering coefficient of the big component are presented in Table 5.4.

Table 5.4: Transitivity and clustering coefficient for the big directed component of `ConceptNet 4`. The first value (NAN) for the clustering coefficient gives the result of the calculation when vertices with less than two neighbors are left out from the calculation, while the second value (ZERO) gives the result of the calculation when vertices with less than two neighbors are considered as having zero transitivity. Note that all values are the same both for directed as well as undirected graphs.

Transitivity	0.000351298700593188
Clustering Coefficient (NAN)	0.098300551193575281
Clustering Coefficient (ZERO)	0.000851478314399245

For information about shortest paths in this component please see Chapter 7.

Figures 5.4, 5.5, 5.6, and 5.7 present the strongly connected components of sizes 3-12.

Component of Size 12

In the strongly connected component of size 12 we can find the concepts `front` (2423), `back` (15583), `side` (17836), `last` (23202), `edge` (24347), `corner` (29067), `after` (31656), `behind` (46824), `middle` (52077), `before` (108544), `rear` (141086), and `centre` (202139). Figure 5.4a presents the induced directed graph of that component.

Component of Size 10

In the first strongly connected component of size 10 we can find the concepts `year` (2709), `week` (2757), `day` (2759), `hour` (2762), `minute` (2764), `night` (8677), `morning` (15749), `afternoon` (15914), `even` (15946), and `month` (25290). Figure 5.4b presents the induced directed graph of that component.

In the second strongly connected component of size 10 we can find the concepts `difficult` (195), `plain` (1155), `soft` (2842), `hard` (7545), `simple` (15368), `easy` (19144), `smooth` (24330), `fancy` (24730), `rough` (34315), and `gentle` (55184). Figure 5.4c presents the induced directed graph of that component.

Component of size 8

In the strongly connected component of size 8 we can find the concepts `cold` (912), `winter` (1431), `summer` (1437), `hot` (1438), `rise` (5930), `heat` (7301), `cool` (7306), and `fall` (9975). Figure 5.4d presents the induced directed graph of that component.

Component of Size 5

In the strongly connected component of size 5 we can find the concepts `local` (60886), `foreigner` (62358), `native` (94333), `express` (141657), and `foreign` (333670). Figure 5.5a presents the induced directed graph of that component.

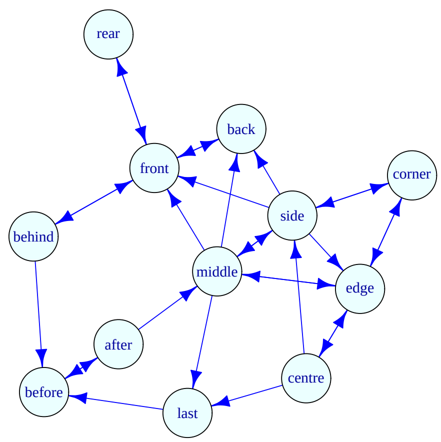
Components of Size 4

In the first strongly connected component of size 4 we can find the concepts `south` (6265), `west` (9659), `north` (22569), and `east` (42579). Figure 5.5b presents the induced directed graph of that component.

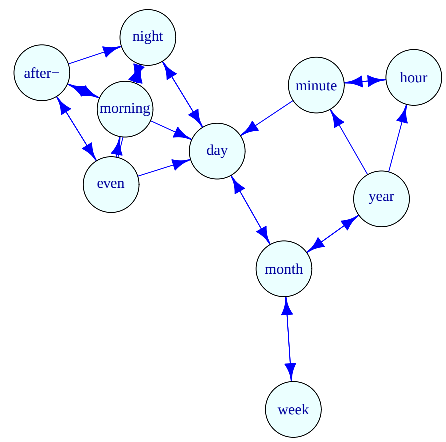
In the second strongly connected component of size 4 we can find the concepts `receive` (15790), `take` (17431), `give` (43731), and `send` (162951). Figure 5.5c presents the induced directed graph of that component.

In the third strongly connected component of size 4 we can find the concepts `sugar` (1446), `salt` (1817), `pepper` (4326), and `spice` (8644). Figure 5.5d presents the induced directed graph of that component.

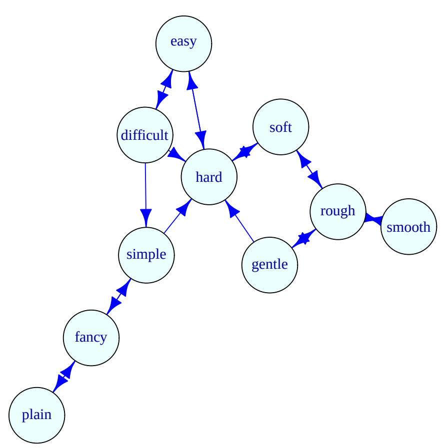
In the fourth strongly connected component of size 4 we can find the concepts `light` (1716), `bright` (1717), `dark` (6376), and `dim` (101382). Figure 5.6a presents the induced directed graph of that component.



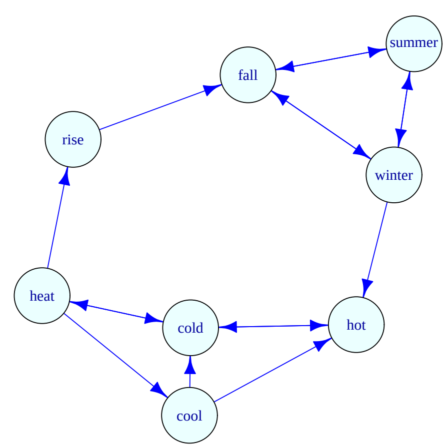
(a) 12 nodes, 29 edges.



(b) 10 nodes, 24 edges.

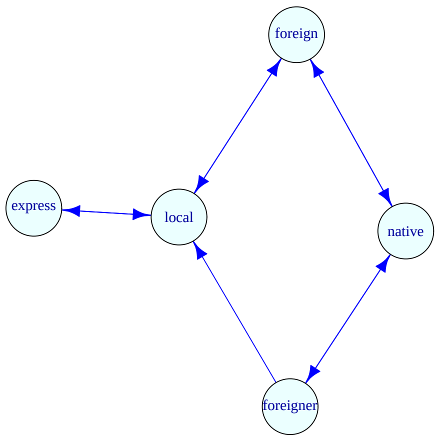


(c) 10 nodes, 20 edges.

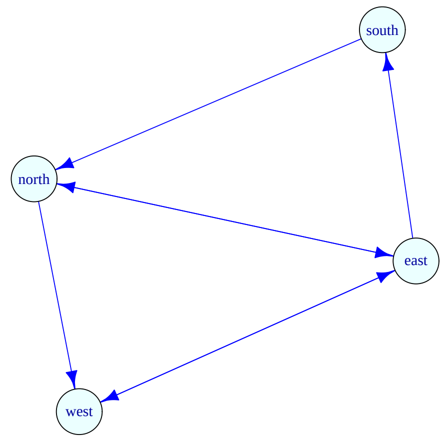


(d) 8 nodes, 16 edges.

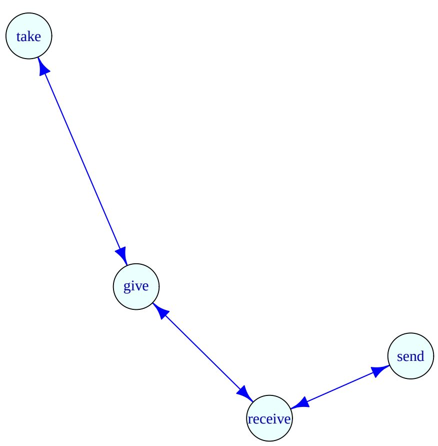
Figure 5.4: The strongly connected components of size 8-12 induced by assertions with negative polarity; see Table 5.3.



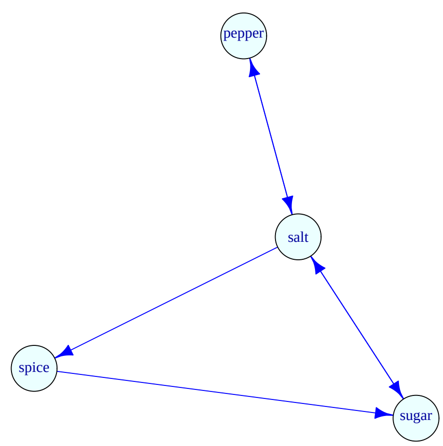
(a) 5 nodes, 9 edges.



(b) 4 nodes, 7 edges.

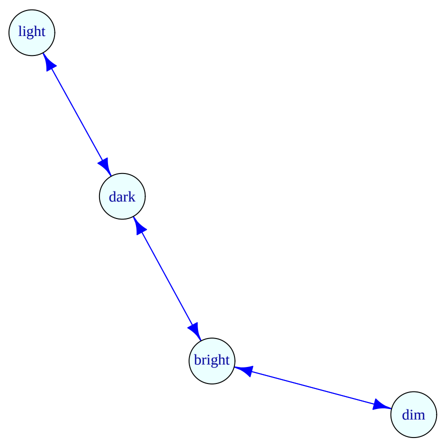


(c) 4 nodes, 6 edges.

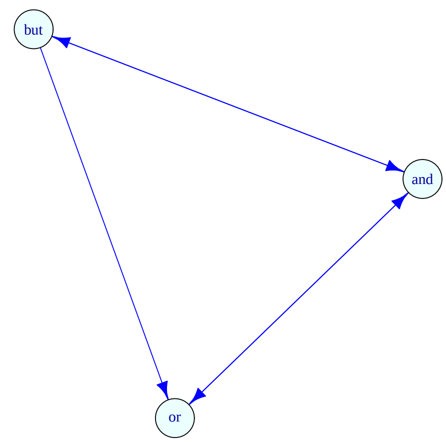


(d) 4 nodes, 6 edges.

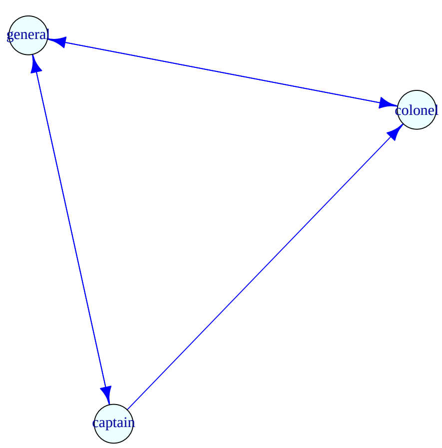
Figure 5.5: Strongly connected components with sizes 4-5 induced by assertions with negative polarity; see Table 5.3.



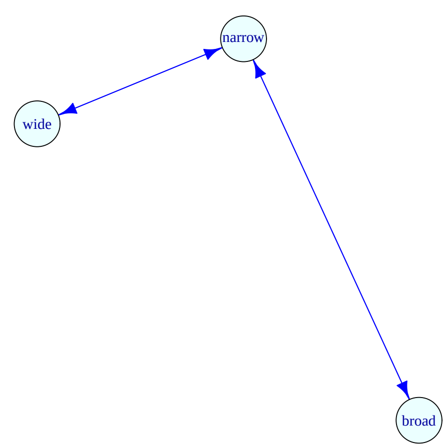
(a) 4 nodes, 6 edges.



(b) 3 nodes, 5 edges.

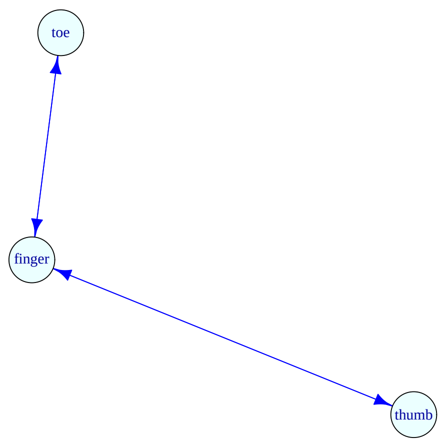


(c) 3 nodes, 5 edges.

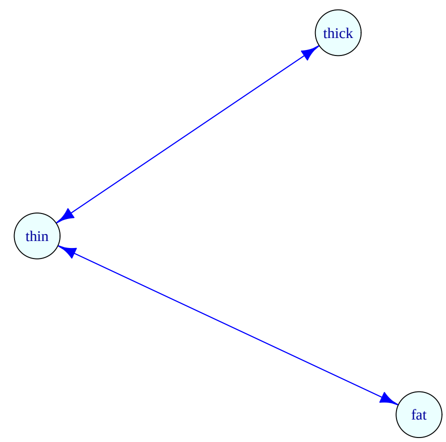


(d) 3 nodes, 4 edges.

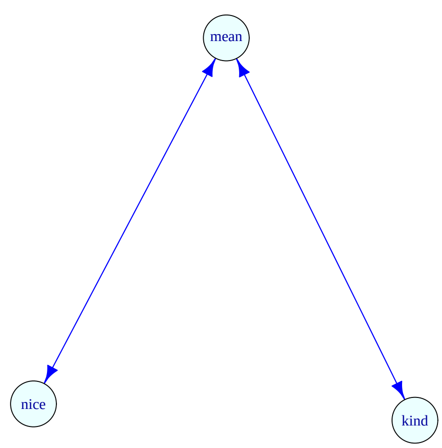
Figure 5.6: The strongly connected components with sizes 3-4 induced by assertions with negative polarity; see Table 5.3.



(a) 4 nodes, 4 edges.



(b) 3 nodes, 4 edges.



(c) 3 nodes, 4 edges.

Figure 5.7: Strongly connected components of size 3 induced by assertions with negative polarity; see Table 5.3.

Components of Size 3

In the first strongly connected component of size 3 we can find the concepts `but` (35882), `and` (40224), and `or` (40776). Figure 5.6b presents the induced directed graph of that component.

In the second strongly connected component of size 3 we can find the concepts `general` (6836), `captain` (23817), and `colonel` (332231). Figure 5.6c presents the induced directed graph of that component.

In the third strongly connected component of size 3 we can find the concepts `narrow` (17316), `wide` (27291), and `broad` (48158). Figure 5.6d presents the induced directed graph of that component.

In the fourth strongly connected component of size 3 we can find the concepts `finger` (3399), `toe` (5571), and `thumb` (15862). Figure 5.7a presents the induced directed graph of that component.

In the fifth strongly connected component of size 3 we can find the concepts `fat` (1763), `thin` (9272), and `thick` (56754). Figure 5.7b presents the induced directed graph of that component.

In the sixth strongly connected component of size 3 we can find the concepts `nice` (2028), `mean` (6744), and `kind` (31540). Figure 5.7c presents the induced directed graph of that component.

5.3 Positive Polarity: Connected Components

In this section we examine the weakly and strongly connected components of the graphs induced by assertions with positive polarity only.

5.3.1 Weakly Connected Components

We get $38,153$ weakly connected components, out of which $22,651$ are isolated vertices. Note that $22,651$ is in complete agreement with Table 3.1. Among the rest $15,502$ components we can find components with cardinalities between 2 and $223,679$.

Distribution of Component Sizes. The distribution of the sizes for the various components is shown in Table 5.5. This distribution presents the cardinalities of the weakly connected components of the induced directed graph, as well as the cardinalities of the connected components of the induced undirected graph. For the induced graphs we consider assertions with positive score in the English language and we allow all frequencies in the edges.

Table 5.5: Distribution of sizes for weakly connected components for the directed graph induced by the assertions with positive polarity only. This is also the distribution of sizes for the connected components of the induced undirected graph.

# of nodes	223,679	55	32	31	30	22	18	16	14	12	11	10	9	8	7	6	5	4	3	2	1
# of components	1	1	1	1	1	2	1	1	4	1	3	3	4	11	14	26	81	196	943	14,207	22,651

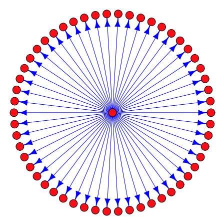
Figure 5.8 presents the weakly connected components with sizes 11-55. Note that in Chapter 7 we will explore the longest geodesic paths of the induced directed and undirected graphs and we will see that that in every case such a path is at least 15. Hence, Figure 5.8 apart from giving an overview of some of the weakly connected components, it also shows that the longest geodesic paths do not come from any of these components.

Big Weakly Connected Component

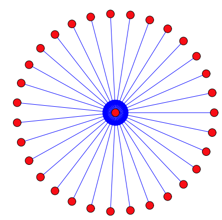
The undirected graph induced by the concepts that appear in the big undirected component is composed of $223,679$ nodes and $383,698$ edges. For information about shortest paths in this component please see Chapter 7.

Component of Size 55

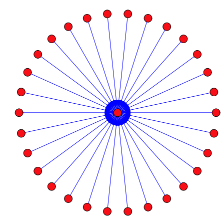
The component of size 55 is a star about *medical specialties*. Concept `medical specialty` (171593) has out-degree 54 and in-degree 0. All the other concepts have out-degree equal to 0 and in-degree equal to 1. These 54 concepts are `concern anesthesia anesthesiology` (171594), `concern bactereia bacteriology` (171595),



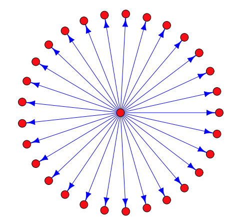
(a) medical specialty
55 nodes, 54 edges.



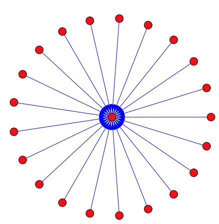
(b) pacific ocean 0 m
32 nodes, 31 edges.



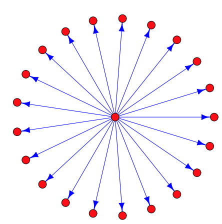
(c) atlantic ocean 0 m
31 nodes, 30 edges.



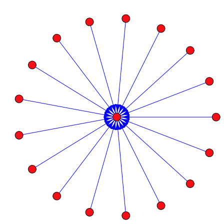
(d) haha
30 nodes, 29 edges.



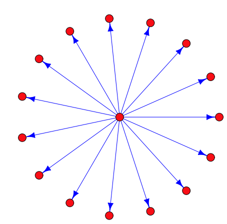
(e) indian ocean 0 m
22 nodes, 21 edges.



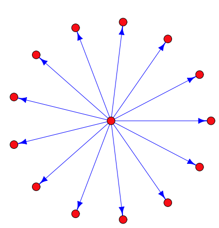
(f) space shuttle acronym
22 nodes, 21 edges.



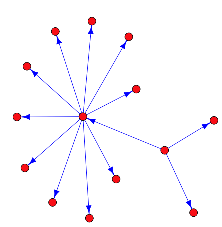
(g) caribbean sea 0 m
18 nodes, 17 edges.



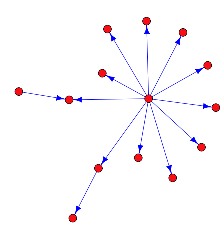
(h) another say safe
16 nodes, 15 edges.



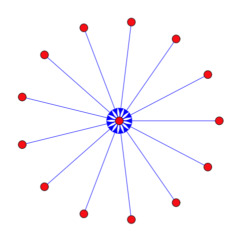
(i) alani
14 nodes, 13 edges.



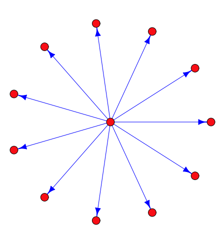
(j) different culture, different country
14 nodes, 13 edges.



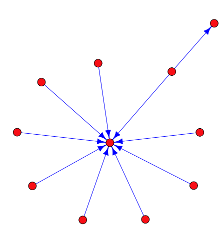
(k) type catheter, two channel, female catheter
14 nodes, 13 edges.



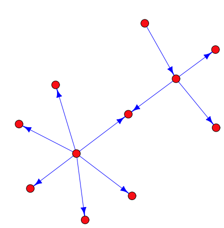
(l) rnum virus
14 nodes, 13 edges.



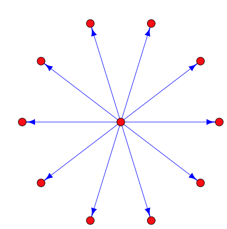
(m) dirge
12 nodes, 11 edges.



(n) darkish region mar, margaritifer sinus
11 nodes, 10 edges.



(o) hydrogen peroxide, h2o2, powerful oxidizer
11 nodes, 10 edges.



(p) sulfa drug
11 nodes, 10 edges.

Figure 5.8: Weakly connected components that arise in the *directed* graph induced by the assertions with positive polarity; see Table 5.5. The names of the subgraphs are given by the nodes with total degrees different from 1. All such nodes are listed in decreasing order of total degree. In case of a tie (Figure 5.8k) precedence takes the name of the node that has larger in-degree (two channel vs. female catheter).

concern birth obstetrics (171596), concern body function physiology (171597), concern body movement kinesiology (171598), concern cell cytology (171599), concern child pediatrics (171600), concern digestive system gastroenterology (171601), concern disease cause etiology (171602), concern disease classification nosology (171603), concern disease identification diagnostic (171604), concern ear otology (171605), concern epidemic epidemiology (171606), concern contagious disease epidemiology (171607), concern eye ophthalmology (171608), concern gland adenology (171609), concern gum periodontics (171610), concern hear audiology (171611), concern heart cardiology (171612), concern hernia herniology (171613), concern intestine entrology (171614), concern joint arthrology (171615), concern joint rheumatology (171616), concern kidney nephrology (171617), concern liver hepetology (171618), concern liver hepatology (171619), concern mental disorder psychiatry (171620), concern mouth stomatology (171621), concern mouth oralogy (171622), concern muscle myology (171623), concern muscle orthopedic (171624), concern nervous system neurology (171625), concern nervous system neurophathology (171626), concern newborn neonatology (171627), concern nose rhinology (171628), concern parasite parasitology (171629), concern poison toxicology (171630), concern toxin toxicology (171631), concern rheumatic disease rheumatology (171632), concern serum serology (171633), concern skin dermatology (171634), concern skull craniology (171635), concern stomach gastrology (171636), concern symptom symptomology (171638), concern tissue histology (171642), concern tumor oncology (171643), concern ulcer helcology (171644), concern vein phlebology (171645), concern virus virology (171646), concern x-ray radiology (171647), concern radiation therapy radiology (171648), concern dentistry tooth (325385), concern tooth straighten orthodontics (325386), and concern tooth dentistry (325387).

Component of Size 32

The component of size 32 is about the *sea level of the pacific ocean*. Concept pacific ocean 0 m (5019) has out-degree 0 and in-degree 31. All the other concepts have out-degree equal to 1 and in-degree equal to 0. These 31 concepts are low point american samoa (5018), low point baker island (5044), low point chile (5082), low point colombia (5086), low point cook island (5088), low point costa rica (5091), low point ecuador (5102), low point el salvador (5107), low point fiji (5117), low point guam (5132), low point guatemala (5133), low point jarvis island (5157), low point kingman reef (5163), low point kiribati (5164), low point marshall island (5187), low point midway island (5197), low point nauru (5220), low point new zealand (5225), low point nicaragua (5226), low point niue (5231), low point norfolk island (5232), low point palau (5241), low point panama (5243), low point peru (5247), low point samoa (5266), low point solomon island (5281), low point tokelau (5306), low point tonga (5308), low point tuvalu (5314), low point vanuatu (5330), and low point wake island (5334).

Component of Size 31

The component of size 31 is about the *sea level of the atlantic ocean*. Concept atlantic ocean 0 m (5022) has out-degree 0 and in-degree 30. All the other concepts have out-degree equal to 1 and in-degree equal to 0. These 30 concepts are low point angola (5021), low point barbados (5046), low point benin (5055), low point bermuda (5056), low point brazil (5064), low point cameroon (5075), low point canada (5076), low point cape verde (5078), low point french guiana (5119), low point gabon (5120), low point ghana (5125), low point greenland (5129), low point guernsey (5134), low point guinea (5136), low point guinea-bissau (5137), low point guyana (5138), low point iceland (5143), low point ireland (5150), low point jersey (5158), low point liberia (5173), low point namibia (5218), low point nigeria (5230), low point portugal (5253), low point saint helena (5263), low point senegal (5270), low point sierra leone (5274), low point south africa (5283), low point spain (5287), low point togo (5305), and low point uruguay (5327).

Component of Size 30

The component of size 30 is about the endangered *haha plant species*. Concept haha (13162) has out-degree 29 and in-degree 0. All the other concepts have out-degree equal to 0 and in-degree equal to 1. These 29 concepts are cyanea acuminata (13163), cyanea asarifolia (13164), cyanea copelandius copelandius (13165), cyanea copelandius haleakalaensis (13166), cyanea crispa (13167), cyanea dunbarius (13168), cyanea grimesiana grimesiana (13172), cyanea grimesiana obata (13174), cyanea hamatiflora hamatiflora (13176), cyanea humboldtiana (13177), cyanea koolauensis (13178), cyanea lobata (13179), cyanea longiflora

(13180), `cyanea mceldowneyi` (13184), `cyanea pinnatifida` (13185), `cyanea platyphylla` (13186), `cyanea procera` (13187), `cyanea recta` (13188), `cyanea remyi` (13189), `cyanea stictophylla` (13192), `cyanea superba` (13193), `cyanea truncata` (13194), `cyanea undulata` (13195), `cyanea glabrum` (311277), `cyanea hamatiflora carlsonie` (311278), `cyanea macrostegia gibsonie` (311279), `cyanea mannie` (311280), `cyanea st-johnie` (311281), and `cyanea shipmannie` (311282).

Components of Size 22

The first component of size 22 is about the *sea level of the indian ocean*. Concept `indian ocean 0 m` (5027) has out-degree 0 and in-degree 21. All the other concepts have out-degree equal to 1 and in-degree equal to 0. These 21 concepts are `low point antarctica` (5026), `low point bangladesh` (5045), `low point christmas island` (5085), `low point comoro` (5087), `low point europa island` (5115), `low point glorioso island` (5127), `low point india` (5144), `low point indonesia` (5147), `low point kenya` (5162), `low point madagascar` (5181), `low point malaysia` (5182), `low point maldive` (5183), `low point mauritius` (5191), `low point mayotte` (5193), `low point mozambique` (5216), `low point pakistan` (5240), `low point reunion` (5256), `low point seychelles` (5273), `low point somalia` (5282), `low point sri lanka` (5288), and `low point tanzania` (5303).

The second component of size 22 is about *space shuttle acronyms*. Concept `space shuttle acronym` (172559) has out-degree 21 and in-degree 0. All the other concepts have out-degree equal to 0 and in-degree equal to 1. These 21 concepts are `adi attitude direction indicator` (172560), `apu auxiliary powewr unit` (172561), `css control stick steer` (172562), `dcm display control module` (172563), `eva extravehicular activity` (172564), `hsus horizontal situation indicator` (172565), `iva intravehicular activity` (172566), `lcc launch control center` (172567), `lo loss signal` (172568), `mcc mission control center` (172569), `meet mission elapse time` (172570), `mlp mobile launch platform` (172571), `mmu man maneuver unit` (172572), `om orbital maneuver system` (172573), `pam payload assist module` (172574), `plss portable life support system` (172575), `rc reaction control system` (172576), `rm remote manipulator system` (172577), `srb solid rocket booster` (172578), `tp thermal protection system` (172579), and `wc waste collection system` (172580).

Component of Size 18

The component of size 18 is about the *sea level of the caribbean sea*. Concept `caribbean sea 0 m` (5024) has out-degree 0 and in-degree 17. All the other concepts have out-degree equal to 1 and in-degree equal to 0. These 17 concepts are `low point anguilla` (5023), `low point aruba` (5033), `low point belize` (5052), `low point cayman island` (5079), `low point cuba` (5093), `low point dominica` (5101), `low point grenada` (5130), `low point guadeloupe` (5131), `low point haiti` (5139), `low point honduras` (5140), `low point jamaica` (5154), `low point martinique` (5190), `low point montserrat` (5213), `low point puerto rico` (5254), `low point saint lucia` (5264), `low point venezuela` (5331), and `low point virgin island` (5333).

Component of Size 16

The component of size 16 is about *saying things in a safe way*. Concept `another say safe` (163403) has out-degree 15 and in-degree 0. All the other concepts have out-degree equal to 0 and in-degree equal to 1. These 15 concepts are `say perfectly safe` (324626), `say absolutely safe` (324627), `say really safe` (324628), `say truly safe` (324629), `say obviously safe` (324630), `say undeniably safe` (324631), `say veritably safe` (324632), `say remarkably safe` (324633), `say notably safe` (324634), `say strikingly safe` (324635), `say markedly safe` (324636), `say eminently safe` (324638), `say greatly safe` (324639), `say vastly safe` (324640), `say hugely safe` (324641).

Components of Size 14

The first component of size 14 is about the plant species which is known in Hawaii as *alani*. Concept `alani` (12772) has out-degree 13 and in-degree 0. All the other concepts have out-degree equal to 0 and in-degree equal to 1. These 13 concepts are `melicope adscenden` (12773), `melicope balloui` (12774), `melicope haupuensis` (12775), `melicope lydgatei` (12777), `melicope mucronulata` (12778), `melicope munroi` (12779), `melicope ovali` (12780), `melicope pallida` (12781), `melicope quadrangularis` (12783), `melicope reflexa` (12784), `melicope zahlbruckneri` (12787), `melicope saint-johnie` (311223), and `melicope knudsenie` (311225).

The second component of size 14 revolves around differences that *different cultures* have. Concept `different culture` (17023) has out-degree 10 and in-degree 1. Concept `different country` (76553) has out-degree 3 and in-degree 0. All the other concepts have out-degree equal to 0 and in-degree equal to 1. These 12 concepts are `different tradition` (46475), `different idea taste beauty` (72184), `different law` (76554), `different form art` (89948), `different tonal scale` (90582), `different type jewelry` (91407), `different value system` (100103), `different currency money` (117726), `different tradition celebrate birthday` (131995), `different concept fairness` (175218), `different custom` (311469), and `different custom talk` (316209).

The third component of size 14 has as central notion different *types of catheters*. Concept `type catheter` (169453) has out-degree 11 and in-degree 0. Concept `two channel` (169456) has out-degree 0 and in-degree 2. Concept `female catheter` (169462) has out-degree 1 and in-degree 1. Concept `double-current catheter` (169457) has out-degree 1 and in-degree 0. All the other 10 concepts have out-degree equal to 0 and in-degree equal to 1. These 10 concepts are `uterine catheter` (169454), `cardiac catheter` (169455), `elbow catheter` (169458), `insert through female urethra` (169463), `dilate laryngeal stricture` (169465), `effect bladder drainage` (169466), `foley catheter` (169470), `itard catheter` (169471), `bozeman catheter` (325202), and `mercy catheter` (325203).

The fourth component of size 14 is about *rnum virus*. Concept `rnum virus` (226834) has out-degree 0 and in-degree 13. All the other 13 concepts have out-degree equal to 1 and in-degree equal to 0. These 13 concepts are `retrovirid` (226833), `arenavirid` (226835), `picornavirid` (226836), `calicivirid` (226837), `bunyavirid` (226838), `orthomyxovirid` (226839), `paramyxovirid` (226840), `rhabdovirid` (226841), `pilovirid` (226842), `togavirid` (226843), `flavivirid` (226844), `coronavirid` (226845), and `reovirid` (226887).

Component of Size 12

The component of size 12 is about the notion of *dirge*. Concept `dirge` (173532) has out-degree 11 and in-degree 0. All the other concepts have out-degree equal to 0 and in-degree equal to 1. These 11 concepts are `slow mournful piece music` (173533), `hymn lamentation grief` (173534), `accompany funeral` (173535), `accompany memorial rite` (173536), `any slow solemn piece music` (173537), `death melody` (357753), `funeral march` (361199), `funeral music` (361200), `funeral song` (361202), `mournful song` (368168), `death song` (384749).

Components of Size 11

The first component of size 11 is about *dark regions on Mars* [1]. Concept `darkish region mar` (106353) has out-degree 0 and in-degree 9. Concept `margaritifer sinus` (106363) has out-degree 2 and in-degree 0. Concept `darkish area mar` (106364) has out-degree 0 and in-degree 1. All the other concepts have out-degree equal to 1 and in-degree equal to 0. These 8 concepts are `nilokera` (106352), `iapygia` (106354), `mare hadriaticum` (106355), `hellespontu` (106356), `propoutis` (106371), `noctis lacus` (106374), `tithonius lacus` (106379), and `chrysokera` (106380).

In the second component of size 11 the concept with the highest degree is *hydrogen peroxide*. Concept `hydrogen peroxide` (122047) has out-degree 6 and in-degree 0. Concept `h2o2` (122044) has out-degree 3 and in-degree 1. Concept `powerful oxidizer` (122048) has out-degree 0 and in-degree 2. Concept `chemical formula hydrogen peroxide` (122043) has out-degree 1 and in-degree 0. All the other concepts have out-degree equal to 0 and in-degree equal to 1. These 7 concepts are `natural metabolite many organism` (122052), `miscible water` (122053), `deodorize bleach agent` (122058), `clear colorless` (122059), `characteristic pungent odor` (122060), `sell water solution` (122061), and `mild disinfectant` (122063).

The third component of size 11 is about *sulfa drug*. Concept `sulfa drug` (171559) has out-degree 10 and in-degree 0. All the other concepts have out-degree equal to 0 and in-degree equal to 1. These 10 concepts are `derive sulfanilamide` (171560), `treat infection` (171561), `treat conjunctivitis` (171562), `treat bronchitis` (171563), `treat leprosy` (171564), `treat malaria` (171565), `treat dysentery` (171566), `treat gastroenteritis` (171567), `treat urinary infection` (171568), and `prevent growth bacterium` (171569).

[1] See also `http://conceptnet5.media.mit.edu/web/c/en/darkish_region_on_mar`.

5.3.2 Strongly Connected Components

We have $265,696$ strongly connected components, out of which $265,596$ are isolated vertices. Among the rest 100 components we can find one component of size $13,700$, three components of size 3, and ninety six components of size 2.

Note that the numbers presented here for strongly connected components refer to the case of the directed graph only since in the undirected case we have the notion of connected components which is the same as the weakly connected components of the directed graph. Those were presented earlier.

Distribution of Component Sizes. The distribution of the sizes for the various components is shown in Table 5.6.

Table 5.6: Distribution of sizes for strongly connected components for the induced directed graph.

# of nodes per component	13,700	3	2	1
# of components	1	3	96	265,596

Big Strongly Connected Component

Regarding the big strongly connected component with the $13,700$ nodes, it has $120,865$ edges (self-loops were omitted from the enumeration). Hence the average degree is about 17.64453 after self-loops have been discarded. Regarding the induced undirected graph that occurs after restricting ourselves in these $13,700$ nodes (again, self-loops are omitted), the number of edges is $109,378$. In other words, the average degree in this case is about 15.96759. The transitivity and the clustering coefficient of the big component are presented in Table 5.7.

Table 5.7: Transitivity and clustering coefficient for the big directed component of `ConceptNet 4`. The first value (NAN) for the clustering coefficient gives the result of the calculation when vertices with less than two neighbors are left out from the calculation, while the second value (ZERO) gives the result of the calculation when vertices with less than two neighbors are considered as having zero transitivity. Note that all values are the same both for directed as well as undirected graphs.

Transitivity	0.0453658181737l4129
Clustering Coefficient (NAN)	0.219425693644797526
Clustering Coefficient (ZERO)	0.195080653182017061

For information about shortest paths in this component please see Chapter 7.

Components of Size 3

The first strongly connected component of size 3 is composed of the concepts `first floor` (1598), `second floor` (9162), and `third floor` (141542).

The second strongly connected component of size 3 is composed of the concepts `primary color` (9707), `red yellow blue` (15197), and `three primary color` (32853).

The third strongly connected component of size 3 is composed of the concepts `capital unite state` (3370), `washington dc` (3371), and `washington d.c` (5028).

5.4 Both Polarities

In this section we examine the weakly and strongly connected components of the directed graph induced by the assertions with both polarities; that is, both negative and positive.

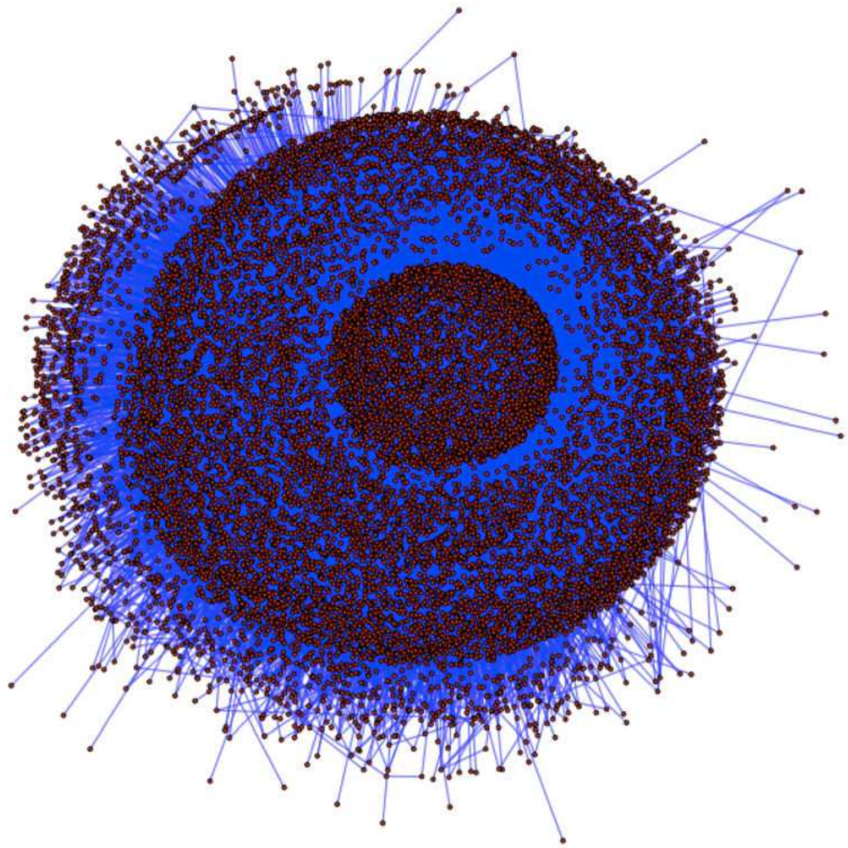


Figure 5.9: The maximal strongly connected component; see Table 5.6. For simplicity we plot the induced undirected graph of that component (in low resolution).

5.4.1 Weakly Connected Components

We get $32,702$ weakly connected components, out of which $16,922$ are isolated vertices. Note that $16,922$ is in complete agreement with Table 3.1. Among the rest $15,780$ components we can find components with cardinalities between 2 and $228,784$.

Distribution of Component Sizes. The distribution of the sizes for the various components is shown in Table 5.8. This distribution presents the cardinalities of the weakly connected components of the induced directed graph, as well as the cardinalities of the connected components of the induced undirected graph. For the induced graphs we consider assertions with positive score in the English language and we allow all frequencies in the edges.

Table 5.8: Distribution of sizes for weakly connected components for the induced directed graph. This is also the distribution of sizes for the connected components of the induced undirected graph.

# of nodes	228,784	55	32	31	30	22	18	16	14	12	11	10	9	8	7	6	5	4	3	2	1
# of components	1	1	1	1	1	2	1	1	4	1	3	2	5	11	16	27	85	204	970	14,443	16,922

Big Weakly Connected Component

The undirected graph induced by the concepts that appear in the big undirected component is composed of $228,784$ nodes and $394,554$ edges. For information about shortest paths in this component please see Chapter 7.

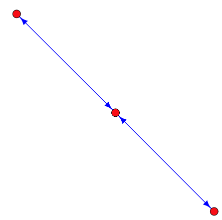


Figure 5.10: The three strongly connected components of size 3 look identical (self-loops have been neglected).

Components of Sizes 11-55

The weakly connected components of sizes 11-55 are precisely the same as those mentioned as weakly connected components that arise in the directed graph induced by the assertions with positive polarity only.

5.4.2 Strongly Connected Components

We have 265,374 strongly connected components, out of which 265,276 are isolated vertices. Among the rest 98 components we can find one component of size 14,025, two components of size 3, and ninety five components of size 2.

Note that the numbers presented here for strongly connected components refer to the case of the directed graph only since in the undirected case we have the notion of connected components which is the same as the weakly connected components of the directed graph and which were presented earlier.

Distribution of Component Sizes. The distribution of the sizes for the various components is shown in Table 5.9.

Table 5.9: Distribution of sizes for strongly connected components for the induced directed graph.

# of nodes per component	14,025	3	2	1
# of components	1	2	95	265,276

Big Strongly Connected Component

Regarding the big strongly connected component with the 14,025 nodes, it has 126,151 edges (self-loops were omitted from the enumeration). Hence the average degree is about 17.98945 after self-loops have been discarded. Regarding the induced undirected graph that occurs after restricting ourselves in these 14,025 nodes (again, self-loops are omitted), the number of edges is 114,294. In other words, the average degree in this case is about 16.29861. The transitivity and the clustering coefficient of the big component are presented in Table 5.10.

For information about shortest paths in this component please see Chapter 7.

Components of Size 3

The first strongly connected component of size 3 is composed of the concepts `first floor` (1598), `second floor` (9162), `third floor` (141542).

The second strongly connected component of size 3 is composed of the concepts `primary color` (9707), `red yellow blue` (15197), `three primary color` (32853).

The figures of these two components of size 3 of course have not changed from the case where they appeared as strongly connected components induced by assertions with positive polarity only. As a reminder, Figure 5.10 presents the components.

Table 5.10: Transitivity and clustering coefficient for the big directed component of `ConceptNet 4`. The first value (NAN) for the clustering coefficient gives the result of the calculation when vertices with less than two neighbors are left out from the calculation, while the second value (ZERO) gives the result of the calculation when vertices with less than two neighbors are considered as having zero transitivity. Note that all values are the same both for directed as well as undirected graphs.

Transitivity	0.042730645545158707
Clustering Coefficient (NAN)	0.228343346540729242
Clustering Coefficient (ZERO)	0.203807630088901875

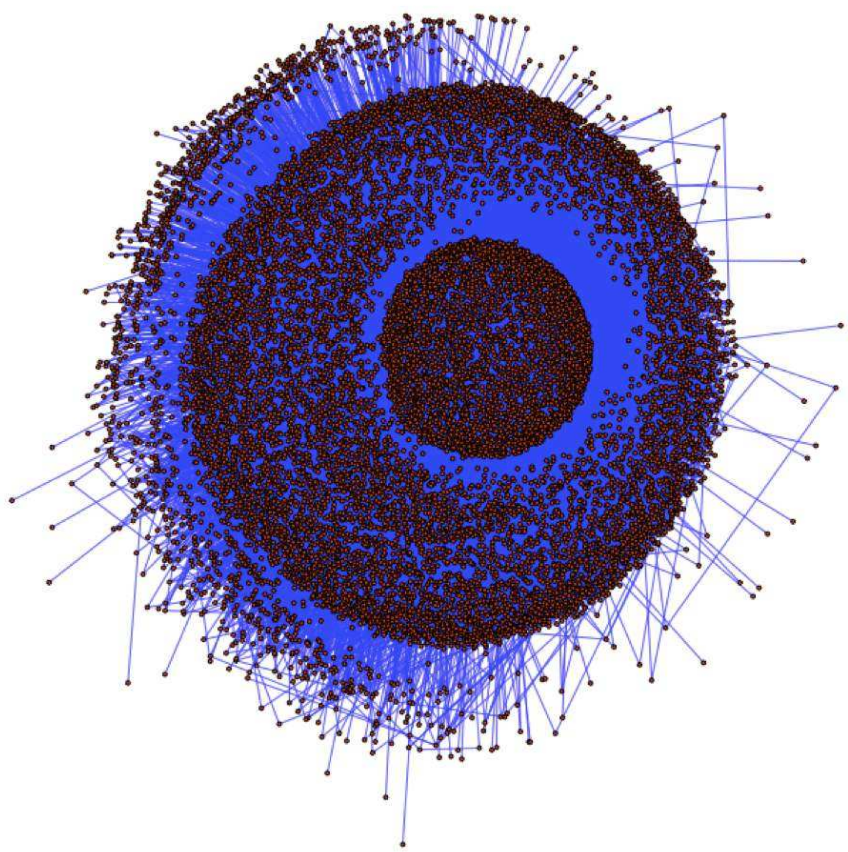


Figure 5.11: The maximal strongly connected component; see Table 5.9. For simplicity we plot the induced undirected graph of that component (in low resolution).

Chapter 6

Cores

We restrict on edges with positive score and allow all frequencies (that is both positive and negative polarity). We distinguish three main cases on whether we allow edges with negative only polarity, positive only polarity, or finally both polarities.

6.1 Negative Polarity

We distinguish cases based on whether we allow self-loops or not.

6.1.1 Loops are Neglected

Table 6.1 presents the distribution of the vertices with specific coreness in the case where self-loops have been neglected. Table 6.2 presents the number of vertices with coreness above a certain threshold, as well as the number of edges and the average degree in every induced graph; whether that is a multigraph, a directed graph, or an undirected graph.

Table 6.1: Distribution of vertices with specific coreness. We only consider assertions with positive score in the English language. The polarity is negative. Self-loops are neglected.

coreness	0	1	2	3	4	5	6
vertices	$267,790$	$9,952$	935	473	172	107	68

Table 6.2: Number of vertices, edges, and the average degree of the induced subgraphs in the case where we allow edges with negative polarity only. Self-loops are neglected.

coreness	vertices	directed multigraph		directed graph		undirected graph	
		edges	avg. degree	edges	avg. degree	edges	avg. degree
$\geqslant 0$	279497	13497	0.096581	13387	0.095794	12989	0.092946
$\geqslant 1$	11707	13497	2.305800	13387	2.287008	12989	2.219014
$\geqslant 2$	1755	4839	5.514530	4747	5.409687	4411	5.026781
$\geqslant 3$	820	3006	7.331707	2930	7.146341	2710	6.609756
$\geqslant 4$	347	1593	9.181556	1540	8.876081	1447	8.340058
$\geqslant 5$	175	911	10.411429	867	9.908571	819	9.360000
$\geqslant 6$	68	348	10.235294	331	9.735294	308	9.058824

The 68 concepts that we find in the innermost core are `person` (9), `tree` (33), `exercise` (61), `library` (68), `bath` (70), `human` (80), `walk` (97), `drink` (120), `examination` (121), `fun` (134), `bed` (156), `park` (365), `talk` (394),

eat (432), computer (467), car (529), dog (537), music (542), cat (616), house (652), fish (655), plant (716), animal (902), bird (962), drive car (1005), desk (1043), office (1044), home (1045), kitchen (1078), eye (1160), die (1227), money (1240), mouse (1284), television (1298), food (1359), horse (1412), hot (1438), read (1456), drive (1545), potato (1674), telephone (1790), audience (1816), rain (1856), book (2033), boat (2389), time (2494), fire (2895), god (4277), space (4435), cabinet (5663), table (5665), long hair (5916), metal (6491), way (6679), competitive activity (7019), ear (8314), gasoline (8502), fly (9215), program language (13345), gerbil (14223), software (17383), brain (17555), cash register (23016), conscious (23506), singular (33174), transportation device (200905), speedo (203600), and fidelity (203658).

6.1.2 Loops are Retained

Table 6.3 presents the distribution of the vertices with specific coreness in the case where self-loops are retained. Table 6.4 presents the number of vertices with coreness above a certain threshold, as well as the number of edges and the average degree in every induced graph; whether that is a multigraph, a directed graph, or an undirected graph.

Table 6.3: Distribution of vertices with specific coreness. We only consider assertions with positive score in the English language. The polarity is negative. Self-loops are retained.

coreness	0	1	2	3	4	5	6
vertices	267,790	9,949	934	477	170	91	86

Table 6.4: Number of vertices, edges, and the average degree of the induced subgraphs in the case where we allow edges with negative polarity only. Self-loops are retained.

coreness	vertices	directed multigraph		directed graph		undirected graph	
		edges	avg. degree	edges	avg. degree	edges	avg. degree
$\geqslant 0$	279497	13510	0.096674	13399	0.095879	13001	0.093031
$\geqslant 1$	11707	13510	2.308021	13399	2.289058	13001	2.221064
$\geqslant 2$	1758	4853	5.521047	4760	5.415245	4424	5.032992
$\geqslant 3$	824	3025	7.342233	2948	7.155340	2727	6.618932
$\geqslant 4$	347	1601	9.227666	1547	8.916427	1454	8.380403
$\geqslant 5$	177	926	10.463277	881	9.954802	833	9.412429
$\geqslant 6$	86	457	10.627907	428	9.953488	401	9.325581

In both cases the maximum coreness is equal to 6. The core in this case contains all the concepts mentioned earlier (case where self-loops were neglected), as well as the concepts man (7), work (35), it (137), child (178), rest (310), housework (343), sleep (425), drawer (495), baby (678), water (1016), see (1161), speak (1305), lie (1395), write (1893), wet (2456), sex (2825), wait (2858), and eye up down (32844).

6.2 Positive Polarity

We distinguish cases based on whether we allow self-loops or not.

6.2.1 Loops are Neglected

Table 6.5 presents the distribution of the vertices with specific coreness in the case where self-loops have been neglected. Table 6.6 presents the number of vertices with coreness above a certain threshold, as well as the number of edges and the average degree in every induced graph; whether that is a multigraph, a directed graph, or an undirected graph.

Table 6.5: Distribution of vertices with specific coreness. We only consider assertions with positive score in the English language. The polarity is positive. Self-loops are neglected.

coreness	0	1	2	3	4	5	6	7	8	9	10	11	12	13
vertices	22651	215187	19847	6948	3381	2091	1488	1154	867	701	548	474	414	339

coreness	14	15	16	17	18	19	20	21	22	23	24	25	26
vertices	302	258	230	233	211	166	142	156	195	156	191	298	869

The 869 concepts that we find in the innermost core are something (5), man (7), person (9), type (11), train (19), town (21), rock (23), beach (24), tree (33), work (35), write program (38), monkey (42), soup (43), go concert (44), hear music (45), weasel (48), word (51), exercise (61), pant (63), love (67), library (68), bath (70), school (73), listen (75), kitten (78), arm (79), human (80), go performance (86), plane (89), class (93), take walk (96), walk (97), entertain (100), run marathon (101), beaver (103), wait line (106), attend lecture (108), drink (120), study (122), go walk (128), play basketball (133), fun (134), it (137), paper (149), bore (152), bed (156), wait table (157), go see film (159), go work (161), watch tv show (163), dirty (170), wake up morning (171), dream (172), shower (173), child (178), smoke (188), chicken (191), go fish (193), state (196), tell story (199), surf web (203), gym (206), play football (209), office build (210), movie (213), wiener dog (220), go restaurant (225), visit museum (228), study subject (234), live life (236), go sport event (241), go play (242), sit (243), play soccer (252), go jog (260), take shower (261), play ball (262), ball (263), eat food (264), watch movie (265), watch film (269), stretch (271), play frisbee (274), go school (276), box (279), object (280), surprise (289), paint picture (291), mother (301), go film (305), party (307), rest (310), listen radio (311), coffee (314), kiss (316), remember (325), candle (327), housework (343), clean (344), lunch (345), street (350), watch tv (351), fungus (354), attend school (355), play tennis (357), park (365), trouble (366), snake (369), wood (370), comfortable (371), play (372), take bus (376), bus (377), conversation (390), talk (394), take course (400), learn (401), plan (408), think (412), go run (423), sleep (425), hang out bar (427), plan vacation (429), go see play (431), eat (432), attend class (433), go swim (442), bridge (444), cloud (446), ride bike (460), nothing (466), computer (467), line (474), buy (475), eat restaurant (479), milk (481), tv (483), stress (486), drawer (495), storage (496), boredom (519), ticket (522), car (529), vehicle (530), dog (537), music (542), zoo (547), use television (560), dress (562), bottle (565), live (580), one (581), turn (583), material (591), chair (596), entertainment (607), cat (616), hat (629), country (640), listen music (642), enjoyment (643), market (648), house (652), fish (655), lake (660), baby (678), hurt (686), hotel (688), plant (716), game (732), hospital (865), bank (867), hide (869), girl (876), student (886), muscle (891), woman (895), animal (902), church (904), cold (912), family (915), go movie (920), moon (924), enlightenment (926), pet (933), cook (946), shop (948), stand line (958), letter (960), bird (962), attend classical concert (972), death (977), play sport (983), eat dinner (984), effort (1000), concert (1001), drive car (1005), bathroom (1007), city (1013), traveling (1014), shark (1015), water (1016), rosebush (1031), yard (1032), knowledge (1040), desk (1043), office (1044), home (1045), sloth (1047), teach (1052), bat (1057), call (1061), couch (1072), kitchen (1078), lizard (1084), laugh joke (1095), run (1102), build (1104), restaurant (1111), spoon (1116), butter (1118), read book (1121), education (1122), beautiful (1124), take note (1136), travel (1143), key (1151), electricity (1153), go store (1157), eye (1160), see (1161), story (1164), nose (1171), smell (1172), stand (1183), well (1201), pen (1205), go sleep (1207), tire (1221), attention (1224), die (1227), fall asleep (1234), money (1240), bill (1245), snow (1247), weather (1248), leg (1252), everything (1262), run errand (1274), patience (1275), mouse (1284), spend money (1286), cry (1291), pay bill (1292), earn money (1293), television (1298), speak (1305), magazine (1310), take bath (1316), hole (1318), nature (1324), band (1330), bald eagle (1331), nest (1332), drink water (1333), crab (1334), paint (1338), ficus (1339), sea (1347), anemone (1348), ocean (1349), sun (1353), sky (1354), fatigue (1357), food (1359), grape (1366), take break (1368), bedroom (1372), hike (1383), drink alcohol (1386), lie (1395), play chess (1398), horse (1412), store (1414), friend (1429), hot (1438), airport (1439), anger (1441), sugar (1446), grocery store (1447), read (1456), curiosity (1460), basket (1463), hold (1464), kill (1466), pay (1473), swim (1475), break (1476), foot (1485), verb (1490), refrigerator (1503), newspaper (1506), rice (1510), drive (1545), surface (1550), liquid (1551), meadow (1558), camp (1566), use computer (1576), window (1577), oil (1587), cover

Table 6.6: Number of vertices, edges, and the average degree of the induced subgraphs in the case where we allow edges with positive polarity only. Self-loops are neglected.

coreness	vertices	directed multigraph		directed graph		undirected graph	
		edges	avg. degree	edges	avg. degree	edges	avg. degree
⩾ 0	279497	478499	3.424001	412956	2.954994	401367	2.872067
⩾ 1	256846	478499	3.725960	412956	3.215592	401367	3.125351
⩾ 2	41659	265649	12.753499	211716	10.164238	201678	9.682326
⩾ 3	21812	220926	20.257290	172260	15.794975	162691	14.917568
⩾ 4	14864	196715	26.468649	151300	20.357912	142112	19.121636
⩾ 5	11483	180655	31.464774	137587	23.963598	128731	22.421144
⩾ 6	9392	168135	35.803876	126962	27.036201	118389	25.210605
⩾ 7	7904	157301	39.802885	117870	29.825405	109561	27.722925
⩾ 8	6750	147343	43.657185	109592	32.471704	101564	30.093037
⩾ 9	5883	138622	47.126296	102469	34.835628	94709	32.197518
⩾ 10	5182	130577	50.396372	95907	37.015438	88462	34.142030
⩾ 11	4634	123595	53.342685	90225	38.940440	83039	35.839016
⩾ 12	4160	116859	56.182212	84803	40.770673	77882	37.443269
⩾ 13	3746	110365	58.924186	79617	42.507742	72978	38.963161
⩾ 14	3407	104665	61.441151	75015	44.035809	68613	40.277664
⩾ 15	3105	98979	63.754589	70526	45.427375	64412	41.489211
⩾ 16	2847	93671	65.803302	66458	46.686336	60583	42.559185
⩾ 17	2617	88669	67.763852	62580	47.825755	56939	43.514712
⩾ 18	2384	83213	69.809564	58343	48.945470	53011	44.472315
⩾ 19	2173	77733	71.544409	54297	49.974229	49265	45.342844
⩾ 20	2007	73342	73.086198	50929	50.751370	46145	45.984056
⩾ 21	1865	69363	74.383914	47915	51.383378	43330	46.466488
⩾ 22	1709	64691	75.706261	44442	52.009362	40099	46.926858
⩾ 23	1514	58327	77.050198	39828	52.612946	35870	47.384412
⩾ 24	1358	52859	77.848306	35945	52.938144	32314	47.590574
⩾ 25	1167	45989	78.815767	30980	53.093402	27810	47.660668
⩾ 26	869	34394	79.157652	22898	52.699655	20526	47.240506

(1592), take film (1595), plate (1604), dinner (1605), smile (1606), den (1610), cow (1613), earth (1633), garage (1647), fiddle (1652), we (1653), garden (1660), wrestle (1665), see new (1666), dance (1667), poop (1672), potato (1674), fight (1675), outside (1676), job (1677), smart (1678), play baseball (1687), frog (1692), napkin (1698), excite (1704), light (1716), salad (1720), fox (1746), forest (1747), attend rock concert (1754), hear news (1758), glass (1776), cupboard (1777), contemplate (1784), telephone (1790), marmot (1796), mountain (1797), pain (1813), audience (1816), salt (1817), motel (1827), drop (1846), bone (1852), meat (1853), bookstore (1854), rain (1856), understand (1858), body (1861), use (1867), ferret (1880), small dog (1882), write (1893), cloth (1903), factory (1917), bottle wine (1918), doll (1931), stay healthy (1932), pencil (1953), research (1978), learn new (1983), wheel (1995), lemur (1998), sweat (2002), name (2003), nice (2028), book (2033), museum (2036), pool (2049), headache (2062), black (2063), canada (2076), fart (2079), instrument (2086), read newspaper (2102), sport (2130), understand better (2163), bad (2226), show (2243), trash (2260), can (2261), a (2263), wind (2284), hand (2300), write story (2335), pee (2354), stop (2358), picture (2360), transportation (2364), road (2368), fall down (2369), seat (2374), boat (2389), wild (2391), practice (2399), help (2410), clothe (2415), dish (2419), train station (2424), lose (2426), war (2438), mall (2447), close eye (2449), wet (2456), flower (2459), wallet (2466), room (2480), satisfaction (2483), time (2494), answer question (2512), perform (2523), cell (2535), small (2536), bicycle (2554), new york (2556), need (2557), farm (2562), sink (2563), pocket (2566), everyone (2589), go somewhere (2592), color (2611), white (2612), red (2614), stone (2631), vegetable (2636), green (2637), life (2638), burn (2644), sound (2660), good (2666), play card (2667), large (2771), shoe (2790), go (2801),

scale (2817), sex (2825), soft (2842), wait (2858), buy ticket (2866), steak (2878), gain knowledge (2890), fire (2895), news (2905), beer (3052), interest (3086), finger (3399), feel (3404), knife (3405), dangerous (3439), sit down (3442), marmoset (3443), carpet (3450), bowl (3463), australia (3494), ski (3524), surf (3525), corn (3531), fridge (3535), soap (3536), expensive (3546), teacher (3556), leave (3571), coin (3573), number (3576), fruit (3590), happiness (3603), exhaustion (3605), sit chair (3608), laugh (3635), heavy (3663), map (3668), fork (3671), cuba (3797), france (3826), italy (3881), steel (3907), piano (4010), wall (4030), club (4076), theatre (4095), unite state (4102), cup (4116), hill (4124), square (4138), relax (4187), apple tree (4194), shelf (4203), waste time (4217), pleasure (4231), relaxation (4254), god (4277), care (4323), friend house (4329), procreate (4344), airplane (4359), watch (4406), space (4435), phone (4517), this (4539), place (4570), radio (4587), tool (4595), apple (4596), mouth (4628), funny (4647), win (4676), go mall (4699), bag (4743), doctor (4760), theater (4770), river (4784), blue (4808), grass (4815), cheese (4844), mammal (4850), bean (4896), lot (4905), hair (4957), flirt (4969), pass time (5077), make (5239), noise (5363), measure (5370), shape (5400), flat (5450), utah (5454), plastic (5505), container (5516), climb (5526), wash hand (5539), go home (5555), bar (5558), bug (5563), view (5574), live room (5581), toilet (5616), love else (5621), tooth (5622), drunk (5628), cabinet (5663), table (5665), furniture (5668), peace (5670), lamp (5671), pizza (5708), sing (5711), buy beer (5734), dust (5736), sand (5768), internet (5811), kid (5854), hall (5865), dictionary (5905), rise (5930), closet (5967), boy (5976), like (5989), date (5999), door (6022), record (6029), find (6040), floor (6062), song (6068), play game (6081), meet (6085), not (6150), activity (6207), basement (6220), sofa (6231), cut (6250), page (6264), company (6274), bite (6368), dark (6376), science (6395), college (6396), world (6404), air (6408), sheep (6424), statue (6436), metal (6491), jog (6511), open (6539), warm (6561), quiet (6583), big (6604), high (6606), squirrel (6609), alcohol (6616), skill (6644), hobby (6671), birthday (6705), university (6708), roll (6734), tiredness (6738), mean (6744), communication (6769), drink coffee (6817), general (6836), clock (6860), read magazine (7049), round (7057), good time (7209), good health (7268), act (7272), play hockey (7283), heat (7301), cool (7306), eat ice cream (7359), learn language (7364), dive (7367), skin (7399), go zoo (7405), go internet (7420), art (7424), noun (7478), top (7514), wine (7522), jar (7524), hard (7545), cash (7584), put (7625), important (7681), duck (7686), toy (7701), ring (7720), read child (7755), crowd (7763), draw (7764), edible (7792), enjoy yourself (7798), wyom (7836), see movie (7891), thing (7936), energy (7982), land (8060), rug (8135), pot (8213), kill person (8251), emotion (8261), little (8268), clean house (8295), change (8313), ear (8314), alive (8379), bread (8404), fit (8548), view video (8571), play poker (8588), excitement (8614), field (8720), move (8737), fly airplane (8753), ride horse (8755), wave (8813), stay bed (8815), look (8821), voice (8828), face (8835), lawn (8860), event (8862), tin (8891), happy (8925), find information (8931), fear (9006), oven (9066), long (9087), go vacation (9089), breathe (9104), shade (9151), carry (9178), recreation (9180), fly (9215), test (9242), enjoy (9244), hear (9269), organization (9275), jump (9278), ride bicycle (9319), egg (9339), building (9384), bee (9700), health (9745), communicate (9747), business (9787), make money (9788), become tire (9805), action (9908), pass (9934), fall (9975), resturant (10012), wash (10170), sock (10193), bear (10208), bell (10210), head (10228), lose weight (10298), jump up down (10301), watch television (10343), sign (10388), count (10461), healthy (10482), end (10507), group (12400), know (13183), pantry (13248), learn subject (13303), bullet (13342), degree (13403), note (13429), card (13442), supermarket (13550), joy (13641), stand up (13725), machine (13790), information (13861), read letter (13879), lay (13886), jump rope (13894), gas (13908), celebrate (13996), roof (14069), brown (14263), circle (14472), cake (14522), solid (15343), dirt (15359), point (15518), useful (15524), handle (15706), adjective (15912), alaska (15970), michigan (15975), maryland (15980), maine (15996), delaware (16177), kansa (16333), department (16725), be (16974), steam (17055), pretty (17204), sadness (17314), bike (17583), side (17836), decoration (18070), watch musician perform (18250), stapler (18341), motion (18365), feel better (18399), classroom (18421), compete (18538), out (18546), feel good (18562), accident (18579), transport (18619), stay fit (18712), injury (18717), ride (18753), play piano (19011), step (19524), apartment (19557), part (19708), bush (19864), course (19871), learn world (19935), countryside (19993), see exhibit (20008), power (20085), same (20650), release energy (20692), see art (20765), see excite story (20985), stage (21403), any large city (21865), comfort (22238), orgasm (22445), trip (22700), laughter (22777), express yourself (23577), discover truth (24279), edge (24347), see favorite show (24507), case (24649), go party (24657), grow (24688), competition (24712), express information (24906), board (24939), climb mountain (24954), attend meet (25060), sunshine (25192), fly kite (25205), examine (25210), race (25233), meet friend (25238), read news (25239), shock (25396), flea (25677), return work (25747), see band (25769), visit art gallery (26118), earn live (26632), punch (26708), cool off (26965), watch television show (27279), socialize (27285), skate (27495), movement (27707), create art (27886),

`crossword puzzle` (28017), `enjoy film` (28066), `go pub` (28343), `feel happy` (28593), `play lacrosse` (28752), `corner` (29067), `socialis` (29314), `away` (29340), `physical activity` (29359), `get` (29712), `short` (30110), `many person` (30864), `outdoors` (30992), `stick` (31425), `singular` (33174), `find house` (33328), `find outside` (34925), `winery` (36809), `branch` (37065), `polish` (38832), `wax` (39314), `make person laugh` (69984), `make friend` (71547), `chat friend` (81516), `meet person` (119411), `meet interest person` (123750), `general term` (172489), `generic` (179027), `ground` (184976), `get drunk` (310177), `eaten` (310995), `friend over` (311108), `get exercise` (311524), `get tire` (311724), `enjoy company friend` (311972), `neighbor house` (312175), `play game friend` (312284), `get physical activity` (312389), `go opus` (312412), `get shape` (312438), `sit quietly` (312805), `do it` (313139), `get fit` (323709), `usually` (328606), `unit` (332537), `generic term` (332695), `teach other person` (427795), `entertain person` (427797), and `see person play game` (427799).

6.2.2 Loops are Retained

Table 6.7 presents the distribution of the vertices with specific coreness in the case where self-loops are retained. Table 6.8 presents the number of vertices with coreness above a certain threshold, as well as the number of edges and the average degree in every induced graph; whether that is a multigraph, a directed graph, or an undirected graph.

Table 6.7: Distribution of vertices with specific coreness. We only consider assertions with positive score in the English language. The polarity is positive. Self-loops are retained.

coreness	0	1	2	3	4	5	6	7	8	9	10	11	12	13
vertices	22649	215183	19841	6955	3383	2091	1486	1156	868	701	548	475	410	339

coreness	14	15	16	17	18	19	20	21	22	23	24	25	26
vertices	302	261	227	234	213	163	142	149	182	170	172	295	902

In both cases the maximum coreness is equal to 26. The core in this case contains all the concepts mentioned earlier (case where self-loops were neglected), as well as the concepts `eat lunch` (969), `buy food` (1068), `eat fast food restaurant` (1407), `football` (1448), `water plant` (1470), `hungry` (1533), `eat breakfast` (1540), `clean room` (1981), `wash clothe` (2121), `suitcase` (2479), `iron` (2587), `idea` (2837), `coat` (4020), `order food` (4424), `eat vegetable` (4895), `touch` (5106), `pray` (5292), `look better` (6191), `wool` (6425), `rabbit` (7815), `clean clothe` (8216), `sneeze` (8538), `analyse` (10415), `taste` (14093), `knit` (14683), `son` (15379), `sense` (18386), `memory` (18563), `inspiration` (18885), `awake` (26369), `butt` (27369), `find truth` (29101), and `stitch` (50513).

6.3 Both Polarities

We distinguish cases based on whether we allow self-loops or not.

6.3.1 Loops are Neglected

Table 6.9 presents the distribution of the vertices with specific coreness in the case where self-loops have been neglected. Table 6.10 presents the number of vertices with coreness above a certain threshold, as well as the number of edges and the average degree in every induced graph; whether that is a multigraph, a directed graph, or an undirected graph.

The 705 concepts that we find in the innermost core are `something` (5), `man` (7), ⟨`censored f-word`⟩ (8), `person` (9), `type` (11), `train` (19), `town` (21), `rock` (23), `beach` (24), `tree` (33), `work` (35), `monkey` (42), `soup` (43), `go concert` (44), `hear music` (45), `weasel` (48), `word` (51), `exercise` (61), `love` (67), `library` (68), `bath` (70), `school` (73), `listen` (75), `kitten` (78), `arm` (79), `human` (80), `go performance` (86), `plane` (89), `class` (93), `take walk` (96), `walk` (97), `entertain` (100), `run marathon` (101), `beaver` (103), `wait line` (106), `attend lecture` (108), `drink` (120), `study` (122), `go walk` (128), `play basketball` (133), `fun` (134), `it` (137), `paper` (149), `bore` (152), `bed` (156), `wait table` (157), `go see film` (159), `go work` (161), `watch tv show` (163), `dirty` (170), `wake up morning` (171), `dream` (172), `shower` (173), `child` (178), `smoke` (188), `chicken` (191), `go fish` (193), `state` (196), `tell story` (199), `surf web` (203), `gym` (206), `play football` (209), `office build`

Table 6.8: Number of vertices, edges, and the average degree of the induced subgraphs in the case where we allow edges with positive polarity only. Self-loops are retained.

coreness	vertices	directed multigraph		directed graph		undirected graph	
		edges	avg. degree	edges	avg. degree	edges	avg. degree
$\geqslant 0$	279497	478879	3.426720	413216	2.956855	401627	2.873927
$\geqslant 1$	256848	478879	3.728890	413216	3.217592	401627	3.127352
$\geqslant 2$	41665	266033	12.770095	211980	10.175447	201942	9.693604
$\geqslant 3$	21824	221325	20.282716	172537	15.811675	162968	14.934751
$\geqslant 4$	14869	197105	26.512207	151564	20.386576	142375	19.150582
$\geqslant 5$	11486	181040	31.523594	137846	24.002438	128989	22.460212
$\geqslant 6$	9395	168517	35.873763	127222	27.082916	118649	25.257903
$\geqslant 7$	7909	157701	39.878872	118143	29.875585	109834	27.774434
$\geqslant 8$	6753	147724	43.750629	109853	32.534577	101825	30.156967
$\geqslant 9$	5885	138997	47.237723	102724	34.910450	94964	32.273237
$\geqslant 10$	5184	130954	50.522377	96163	37.099923	88718	34.227623
$\geqslant 11$	4636	123983	53.487058	90485	39.035807	83297	35.934858
$\geqslant 12$	4161	117232	56.347993	85052	40.880558	78130	37.553473
$\geqslant 13$	3751	110801	59.078113	79917	42.611037	73273	39.068515
$\geqslant 14$	3412	105105	61.609027	75313	44.145955	68909	40.392145
$\geqslant 15$	3110	99411	63.929904	70829	45.549196	64711	41.614791
$\geqslant 16$	2849	94086	66.048438	66719	46.836785	60840	42.709723
$\geqslant 17$	2622	89162	68.010679	62895	47.974828	57244	43.664378
$\geqslant 18$	2388	83643	70.052764	58639	49.111390	53301	44.640704
$\geqslant 19$	2175	78126	71.840000	54557	50.167356	49521	45.536552
$\geqslant 20$	2012	73836	73.395626	51253	50.947316	46459	46.181909
$\geqslant 21$	1870	69849	74.704813	48240	51.593583	43646	46.680214
$\geqslant 22$	1721	65389	75.989541	44928	52.211505	40560	47.135386
$\geqslant 23$	1539	59479	77.295647	40643	52.817414	36627	47.598441
$\geqslant 24$	1369	53549	78.230825	36419	53.205259	32764	47.865595
$\geqslant 25$	1197	47392	79.184628	31966	53.410192	28715	47.978279
$\geqslant 26$	902	35985	79.789357	23980	53.170732	21509	47.691796

(210), movie (213), wiener dog (220), visit museum (228), live life (236), go play (242), sit (243), play soccer (252), go jog (260), take shower (261), ball (263), watch movie (265), watch film (269), stretch (271), play frisbee (274), go school (276), box (279), object (280), surprise (289), mother (301), go film (305), party (307), rest (310), listen radio (311), coffee (314), kiss (316), remember (325), housework (343), clean (344), lunch (345), street (350), watch tv (351), fungus (354), attend school (355), play tennis (357), park (365), trouble (366), snake (369), wood (370), play (372), take bus (376), bus (377), conversation (390), talk (394), learn (401), plan (408), think (412), go run (423), sleep (425), hang out bar (427), go see play (431), eat (432), attend class (433), bridge (444), cloud (446), ride bike (460), nothing (466), computer (467), line (474), buy (475), milk (481), tv (483), stress (486), drawer (495), boredom (519), ticket (522), car (529), vehicle (530), dog (537), music (542), zoo (547), use television (560), dress (562), bottle (565), live (580), one (581), turn (583), material (591), chair (596), entertainment (607), cat (616), hat (629), country (640), listen music (642), enjoyment (643), market (648), house (652), fish (655), lake (660), baby (678), hurt (686), hotel (688), plant (716), game (732), hospital (865), bank (867), girl (876), student (886), muscle (891), woman (895), animal (902), church (904), cold (912), family (915), go movie (920), moon (924), pet (933), cook (946), shop (948), stand line (958), letter (960), bird (962), attend classical concert (972), death (977), play sport (983), concert (1001), drive car (1005), bathroom (1007), city (1013), traveling (1014), water (1016), yard (1032), knowledge (1040), desk (1043), office (1044), home (1045), sloth (1047), teach (1052), bat (1057), call (1061), couch (1072), kitchen (1078), lizard (1084), run (1102), build (1104), restaurant (1111), butter (1118), read book (1121), education (1122), beautiful

Table 6.9: Distribution of vertices with specific coreness. We only consider assertions with positive score in the English language. The polarity can be anything. Self-loops are neglected.

coreness	0	1	2	3	4	5	6	7	8	9	10	11	12	13
vertices	16922	219999	20265	7122	3429	2151	1520	1138	893	713	545	492	416	358

coreness	14	15	16	17	18	19	20	21	22	23	24	25	26	27
vertices	293	265	258	219	233	172	140	150	196	133	180	166	424	705

(1124), take note (1136), travel (1143), key (1151), electricity (1153), go store (1157), eye (1160), see (1161), story (1164), nose (1171), smell (1172), stand (1183), pen (1205), go sleep (1207), tire (1221), die (1227), fall asleep (1234), money (1240), bill (1245), snow (1247), leg (1252), everything (1262), patience (1275), mouse (1284), spend money (1286), cry (1291), television (1298), speak (1305), magazine (1310), hole (1318), nature (1324), bald eagle (1331), nest (1332), drink water (1333), crab (1334), paint (1338), ficus (1339), sea (1347), ocean (1349), sun (1353), sky (1354), fatigue (1357), food (1359), grape (1366), take break (1368), bedroom (1372), hike (1383), lie (1395), play chess (1398), horse (1412), store (1414), friend (1429), hot (1438), airport (1439), anger (1441), sugar (1446), grocery store (1447), read (1456), curiosity (1460), basket (1463), hold (1464), kill (1466), pay (1473), swim (1475), break (1476), foot (1485), verb (1490), refrigerator (1503), newspaper (1506), rice (1510), drive (1545), surface (1550), liquid (1551), meadow (1558), camp (1566), use computer (1576), window (1577), oil (1587), cover (1592), take film (1595), plate (1604), dinner (1605), smile (1606), den (1610), cow (1613), earth (1633), garage (1647), we (1653), garden (1660), see new (1666), dance (1667), potato (1674), fight (1675), outside (1676), job (1677), play baseball (1687), napkin (1698), light (1716), salad (1720), fox (1746), forest (1747), hear news (1758), glass (1776), cupboard (1777), telephone (1790), marmot (1796), mountain (1797), pain (1813), audience (1816), salt (1817), motel (1827), drop (1846), bone (1852), meat (1853), bookstore (1854), rain (1856), understand (1858), body (1861), use (1867), ferret (1880), small dog (1882), write (1893), cloth (1903), bottle wine (1918), doll (1931), pencil (1953), research (1978), learn new (1983), wheel (1995), sweat (2002), nice (2028), book (2033), museum (2036), headache (2062), black (2063), canada (2076), fart (2079), read newspaper (2102), sport (2130), bad (2226), show (2243), trash (2260), wind (2284), hand (2300), write story (2335), stop (2358), picture (2360), transportation (2364), road (2368), fall down (2369), seat (2374), boat (2389), practice (2399), help (2410), clothe (2415), dish (2419), train station (2424), lose (2426), war (2438), mall (2447), wet (2456), flower (2459), wallet (2466), room (2480), time (2494), answer question (2512), perform (2523), cell (2535), small (2536), bicycle (2554), new york (2556), need (2557), farm (2562), pocket (2566), everyone (2589), go somewhere (2592), color (2611), white (2612), red (2614), stone (2631), vegetable (2636), green (2637), life (2638), burn (2644), sound (2660), good (2666), play card (2667), large (2771), shoe (2790), go (2801), scale (2817), sex (2825), wait (2858), buy ticket (2866), steak (2878), gain knowledge (2890), fire (2895), beer (3052), interest (3086), finger (3399), feel (3404), knife (3405), dangerous (3439), sit down (3442), carpet (3450), bowl (3463), australia (3494), ski (3524), corn (3531), fridge (3535), soap (3536), expensive (3546), leave (3571), coin (3573), number (3576), fruit (3590), happiness (3603), sit chair (3608), laugh (3635), heavy (3663), map (3668), philippine (3998), wall (4030), theatre (4095), unite state (4102), cup (4116), hill (4124), square (4138), relax (4187), apple tree (4194), shelf (4203), pleasure (4231), relaxation (4254), god (4277), care (4323), friend house (4329), procreate (4344), airplane (4359), watch (4406), space (4435), phone (4517), this (4539), place (4570), radio (4587), tool (4595), apple (4596), mouth (4628), win (4676), go mall (4699), bag (4743), doctor (4760), theater (4770), river (4784), blue (4808), grass (4815), cheese (4844), mammal (4850), lot (4905), hair (4957), flirt (4969), pass time (5077), make (5239), noise (5363), shape (5400), flat (5450), utah (5454), plastic (5505), container (5516), climb (5526), bar (5558), bug (5563), live room (5581), drunk (5628), cabinet (5663), table (5665), furniture (5668), pizza (5708), sing (5711), dust (5736), sand (5768), kid (5854), hall (5865), closet (5967), boy (5976), like (5989), date (5999), door (6022), record (6029), find (6040), floor (6062), song (6068), play game (6081), meet (6085), not (6150), ice cream (6157), activity (6207), basement (6220), storm (6222), sofa (6231), cut (6250), page (6264), company (6274), dark (6376), science (6395), college (6396), world (6404), air (6408), statue (6436), metal (6491), jog (6511), open (6539), warm (6561), big (6604), squirrel (6609), alcohol (6616), skill (6644), hobby (6671), university (6708), roll (6734), communication

Table 6.10: Number of vertices, edges, and the average degree of the induced subgraphs in the case where we allow edges with any polarity. Self-loops are neglected.

coreness	vertices	directed multigraph		directed graph		undirected graph	
		edges	avg. degree	edges	avg. degree	edges	avg. degree
⩾ 0	279497	491996	3.520582	424525	3.037779	412569	2.952225
⩾ 1	262575	491996	3.747470	424525	3.233552	412569	3.142485
⩾ 2	42576	274660	12.902104	218759	10.276165	208346	9.787016
⩾ 3	22311	229111	20.537941	178510	16.001972	168552	15.109318
⩾ 4	15189	204384	26.912107	157027	20.676411	147455	19.416025
⩾ 5	11760	188168	32.001361	143145	24.344388	133885	22.769558
⩾ 6	9609	175298	36.486211	132209	27.517744	123250	25.653034
⩾ 7	8089	164224	40.604277	122932	30.394857	114234	28.244282
⩾ 8	6951	154411	44.428428	114757	33.018846	106341	30.597324
⩾ 9	6058	145511	48.039287	107443	35.471443	99289	32.779465
⩾ 10	5345	137380	51.405051	100778	37.709261	92940	34.776427
⩾ 11	4800	130499	54.374583	95130	39.637500	87548	36.478333
⩾ 12	4308	123564	57.364903	89514	41.557103	82188	38.155989
⩾ 13	3892	116967	60.106372	84290	43.314491	77254	39.698869
⩾ 14	3534	111010	62.823995	79440	44.957555	72638	41.108093
⩾ 15	3241	105484	65.093490	75108	46.348658	68570	42.314101
⩾ 16	2976	100050	67.237903	70901	47.648522	64627	43.432124
⩾ 17	2718	94354	69.428992	66510	48.940397	60533	44.542311
⩾ 18	2499	89258	71.434974	62555	50.064026	56844	45.493397
⩾ 19	2266	83297	73.518976	58094	51.274492	52699	46.512798
⩾ 20	2094	78680	75.148042	54598	52.147087	49464	47.243553
⩾ 21	1954	74761	76.520983	51624	52.839304	46693	47.792221
⩾ 22	1804	70232	77.862528	48259	53.502217	43580	48.314856
⩾ 23	1608	63809	79.364428	43626	54.261194	39333	48.921642
⩾ 24	1475	59301	80.408136	40348	54.709153	36302	49.223051
⩾ 25	1295	52717	81.416216	35686	55.113514	32054	49.504247
⩾ 26	1129	46745	82.807795	31213	55.293180	27972	49.551816
⩾ 27	705	29273	83.043972	19212	54.502128	17144	48.635461

(6769), `general` (6836), `clock` (6860), `competitive activity` (7019), `read magazine` (7049), `round` (7057), `good time` (7209), `act` (7272), `play hockey` (7283), `heat` (7301), `cool` (7306), `dive` (7367), `go zoo` (7405), `art` (7424), `noun` (7478), `wine` (7522), `jar` (7524), `hard` (7545), `put` (7625), `important` (7681), `duck` (7686), `toy` (7701), `ring` (7720), `crowd` (7763), `draw` (7764), `edible` (7792), `see movie` (7891), `thing` (7936), `energy` (7982), `land` (8060), `rug` (8135), `kill person` (8251), `emotion` (8261), `change` (8313), `ear` (8314), `alive` (8379), `bread` (8404), `fit` (8548), `view video` (8571), `play poker` (8588), `excitement` (8614), `field` (8720), `move` (8737), `fly airplane` (8753), `ride horse` (8755), `wave` (8813), `look` (8821), `voice` (8828), `face` (8835), `happy` (8925), `find information` (8931), `fear` (9006), `oven` (9066), `long` (9087), `go vacation` (9089), `breathe` (9104), `shade` (9151), `carry` (9178), `recreation` (9180), `fly` (9215), `enjoy` (9244), `hear` (9269), `jump` (9278), `ride bicycle` (9319), `egg` (9339), `building` (9384), `bee` (9700), `health` (9745), `communicate` (9747), `business` (9787), `make money` (9788), `action` (9908), `pass` (9934), `fall` (9975), `resturant` (10012), `wash` (10170), `sock` (10193), `bear` (10208), `head` (10228), `jump up down` (10301), `watch television` (10343), `sign` (10388), `count` (10461), `know` (13183), `pantry` (13248), `learn subject` (13303), `degree` (13403), `note` (13429), `card` (13442), `supermarket` (13550), `joy` (13641), `stand up` (13725), `machine` (13790), `information` (13861), `lay` (13886), `jump rope` (13894), `gas` (13908), `celebrate` (13996), `gerbil` (14223), `brown` (14263), `circle` (14472), `cake` (14522), `dirt` (15359), `son` (15379), `adjective` (15912), `michigan` (15975), `maine` (15996), `kansa` (16333), `be` (16974), `steam` (17055), `pretty` (17204), `sadness` (17314), `software` (17383), `decoration` (18070), `watch musician perform` (18250), `stapler` (18341), `motion` (18365), `classroom` (18421), `out` (18546), `feel good` (18562), `accident` (18579), `transport`

(18619), injury (18717), ride (18753), play piano (19011), step (19524), apartment (19557), part (19708), learn world (19935), countryside (19993), see exhibit (20008), same (20650), release energy (20692), see art (20765), stage (21403), any large city (21865), comfort (22238), orgasm (22445), trip (22700), laughter (22777), see favorite show (24507), case (24649), go party (24657), grow (24688), competition (24712), board (24939), climb mountain (24954), fly kite (25205), examine (25210), meet friend (25238), visit art gallery (26118), cool off (26965), watch television show (27279), socialize (27285), skate (27495), movement (27707), crossword puzzle (28017), enjoy film (28066), play lacrosse (28752), corner (29067), away (29340), physical activity (29359), get (29712), short (30110), outdoors (30992), stick (31425), singular (33174), make friend (71547), chat friend (81516), meet person (119411), general term (172489), ground (184976), eaten (310995), friend over (311108), get exercise (311524), get tire (311724), enjoy company friend (311972), opus (311995), neighbor house (312175), play game friend (312284), go opus (312412), sit quietly (312805), usually (328606), entertain person (427797), and see person play game (427799).

6.3.2 Loops are Retained

Table 6.11 presents the distribution of the vertices with specific coreness in the case where self-loops are retained. Table 6.12 presents the number of vertices with coreness above a certain threshold, as well as the number of edges and the average degree in every induced graph; whether that is a multigraph, a directed graph, or an undirected graph.

Table 6.11: Distribution of vertices with specific coreness. We only consider assertions with positive score in the English language. The polarity can be anything. Self-loops are retained.

coreness	0	1	2	3	4	5	6	7	8	9	10	11	12	13
vertices	16920	219994	20259	7130	3431	2152	1517	1140	895	711	545	495	413	356

coreness	14	15	16	17	18	19	20	21	22	23	24	25	26	27
vertices	292	269	256	219	234	173	136	139	201	139	174	166	258	883

In both cases the maximum coreness is equal to 27. The core in this case contains all the concepts mentioned earlier (case where self-loops were neglected), as well as the concepts write program (38), pant (63), examination (121), study subject (234), go sport event (241), eat food (264), paint picture (291), candle (327), take course (400), storage (496), sometimes (526), gun (635), hide (869), enlightenment (926), effort (1000), shark (1015), rosebush (1031), laugh joke (1095), spoon (1116), well (1201), weather (1248), dead (1279), take bath (1316), purse (1322), anemone (1348), drink alcohol (1386), football (1448), water plant (1470), ice (1634), fiddle (1652), wrestle (1665), poop (1672), smart (1678), frog (1692), excite (1704), contemplate (1784), subway (1804), factory (1917), stay healthy (1932), lemur (1998), name (2003), pool (2049), instrument (2086), understand better (2163), can (2261), a (2263), sweet (2330), pee (2354), candy (2386), close eye (2449), suitcase (2479), satisfaction (2483), problem (2500), math (2506), sink (2563), iron (2587), cookie (2595), idea (2837), soft (2842), news (2905), surf (3525), teacher (3556), exhaustion (3605), fork (3671), planet (3683), france (3826), italy (3881), steel (3907), piano (4010), coat (4020), waste time (4217), mind (4432), funny (4647), eat vegetable (4895), bean (4896), touch (5106), pray (5292), measure (5370), view (5574), toilet (5616), program (5620), love else (5621), tooth (5622), disease (5645), peace (5670), lamp (5671), often (5700), buy beer (5734), internet (5811), dictionary (5905), rise (5930), bite (6368), sheep (6424), wool (6425), quiet (6583), high (6606), birthday (6705), reproduce (6721), mean (6744), drink coffee (6817), freezer (6822), good health (7268), eat ice cream (7359), learn language (7364), skin (7399), top (7514), cash (7584), leather (7629), read child (7755), rabbit (7815), pot (8213), clean clothe (8216), little (8268), clean house (8295), stay bed (8815), lawn (8860), event (8862), tin (8891), test (9242), seed (9375), cotton (9729), become tire (9805), lip (9870), lose weight (10298), healthy (10482), end (10507), group (12400), bullet (13342), melt (13459), roof (14069), taste (14093), organ (14628), solid (15343), point (15518), useful (15524), handle (15706), alaska (15970), department (16725), brain (17555), side (17836), chocolate (18107), sense (18386), feel better (18399), compete (18538), memory (18563), stay fit (18712), bush (19864), course (19871), power (20085), edge (24347), express

Table 6.12: Number of vertices, edges, and the average degree of the induced subgraphs in the case where we allow edges with any polarity. Self-loops are retained.

coreness	vertices	directed multigraph		directed graph		undirected graph	
		edges	avg. degree	edges	avg. degree	edges	avg. degree
$\geqslant 0$	279497	492389	3.523394	424790	3.039675	412834	2.954121
$\geqslant 1$	262577	492389	3.750435	424790	3.235546	412834	3.144480
$\geqslant 2$	42583	275058	12.918676	219029	10.287157	208616	9.798088
$\geqslant 3$	22324	229524	20.562982	178793	16.018008	168835	15.125873
$\geqslant 4$	15194	204786	26.956167	157295	20.704884	147722	19.444781
$\geqslant 5$	11763	188565	32.060699	143408	24.382896	134147	22.808297
$\geqslant 6$	9611	175686	36.559359	132468	27.565914	123509	25.701592
$\geqslant 7$	8094	164638	40.681492	123210	30.444774	114512	28.295528
$\geqslant 8$	6954	154806	44.522865	115022	33.080817	106606	30.660339
$\geqslant 9$	6059	145891	48.156792	107695	35.548770	99541	32.857237
$\geqslant 10$	5348	137781	51.526178	101049	37.789454	93210	34.857891
$\geqslant 11$	4803	130911	54.512180	95405	39.727254	87820	36.568811
$\geqslant 12$	4308	123932	57.535747	89755	41.668988	82429	38.267874
$\geqslant 13$	3895	117384	60.274198	84570	43.424904	77531	39.810526
$\geqslant 14$	3539	111471	62.995762	79745	45.066403	72940	41.220684
$\geqslant 15$	3247	105947	65.258392	75430	46.461349	68888	42.431783
$\geqslant 16$	2978	100447	67.459369	71165	47.793821	64889	43.578912
$\geqslant 17$	2722	94828	69.675239	66810	49.088905	60827	44.692873
$\geqslant 18$	2503	89701	71.674790	62857	50.225330	57139	45.656412
$\geqslant 19$	2269	83728	73.801675	58377	51.456148	52977	46.696342
$\geqslant 20$	2096	79106	75.482824	54865	52.352099	49726	47.448473
$\geqslant 21$	1960	75291	76.827551	51981	53.041837	47035	47.994898
$\geqslant 22$	1821	71103	78.092257	48864	53.667216	44154	48.494234
$\geqslant 23$	1620	64538	79.676543	44126	54.476543	39808	49.145679
$\geqslant 24$	1481	59875	80.857529	40707	54.972316	36644	49.485483
$\geqslant 25$	1307	53462	81.808722	36189	55.377200	32534	49.784239
$\geqslant 26$	1141	47526	83.305872	31729	55.616126	28456	49.879053
$\geqslant 27$	883	37043	83.902605	24470	55.424689	21918	49.644394

`information` (24906), `sunshine` (25192), `race` (25233), `flea` (25677), `return work` (25747), `earn live` (26632), `punch` (26708), `butt` (27369), `create art` (27886), `many person` (30864), `find house` (33328), `find outside` (34925), `winery` (36809), `branch` (37065), `polish` (38832), `wax` (39314), `slip` (47533), `agent` (58122), `slope` (64669), `make person laugh` (69984), `generic` (179027), `speedo` (203600), `get physical activity` (312389), `get shape` (312438), `do it` (313139), `get fit` (323709), `generic term` (332695), and `teach other person` (427795).

Chapter 7

Shortest Paths

In this chapter we examine properties of the shortest paths found in `ConceptNet 4`.

7.1 Average Shortest Path Lengths

In this section we examine the average path lengths in `ConceptNet 4`, both for the entire graphs as well as for the big connected components that arise. The number of vertices in every case is $279,497$. Recall from Chapter 5 that the entire graphs either by allowing assertions with negative only polarity, or positive only polarity, or both, are disconnected. Hence, in the calculation of the average path lengths we compute the average of the shortest paths within the components; that is, the average of all the pairs of vertices that can be reached by at least one path.

7.1.1 Negative Polarity

Regarding the graph induced by the assertions of the English language with positive score and negative polarity we can observe the following. The average path length for the directed graph is about 6.737. The average path length for the undirected graph is about 3.863. As a reminder, the number of edges of the directed graph (self-loops are omitted) is $13,387$, while the number of edges of the undirected graph (again omitting self-loops) is $12,989$.

Big Weakly Connected Component

The average path length of the big weakly connected component found in the graph induced by the assertions of the English language with positive score and negative polarity is about 3.864.

Big Strongly Connected Component

The average path length of the big strongly connected component found in the graph induced by the assertions of the English language with positive score and negative polarity is about 6.428. If we consider the same component as an undirected graph, then the average path length is about 3.537.

7.1.2 Positive Polarity

Regarding the graph induced by the assertions of the English language with positive score and positive polarity we can observe the following. The average path length for the directed graph is about 4.811. The average path length for the undirected graph is about 4.330. As a reminder, the number of edges of the directed graph (self-loops are omitted) is $412,956$, while the number of edges of the undirected graph (again omitting self-loops) is $401,367$.

Big Weakly Connected Component

The average path length of the big weakly connected component found in the graph induced by the assertions of the English language with positive score and positive polarity is about 4.330.

Big Strongly Connected Component

The average path length of the big strongly connected component found in the graph induced by the assertions of the English language with positive score and positive polarity is about 4.205. If we consider the same component as an undirected graph, then the average path length is about 3.337.

7.1.3 Both Polarities

Regarding the graph induced by the assertions of the English language with positive score and any polarity we can observe the following. The average path length for the directed graph is about 4.772. The average path length for the undirected graph is about 4.280. As a reminder, the number of edges of the directed graph (self-loops are omitted) is $424,525$, while the number of edges of the undirected graph (again omitting self-loops) is $412,569$.

Big Weakly Connected Component

The average path length of the big weakly connected component found in the graph induced by the assertions of the English language with positive score and any polarity is about 4.280.

Big Strongly Connected Component

The average path length of the big strongly connected component found in the graph induced by the assertions of the English language with positive score and any polarity is about 4.167. If we consider the same component as an undirected graph, then the average path length is about 3.291.

7.2 Path Length Distributions

In this section we examine the distributions of the shortest path lengths in `ConceptNet 4`, both for the entire graph, as well as the big connected components that arise in every case. Again we distinguish cases based on the polarity that we allow on the edges.

7.2.1 Negative Polarity

Table 7.1 gives the distribution of the shortest paths in the directed and undirected graph induced by the assertions of the English language with positive score and negative polarity. It also presents the number of pairs for which the second vertex is unreachable from the first one.

Negative Polarity: Big Weakly Connected Component

First we examine the big weakly connected component that arises in the graph induced by the assertions of the English language with positive score and negative polarity. The component has $8,596$ nodes and $11,247$ undirected edges. Table 7.2 gives the distribution of shortest path lengths in this big undirected component.

Negative Polarity: Big Strongly Connected Component

Next we examine the big strongly connected component that arises in the graph induced by the assertions of the English language with positive score and negative polarity. The component has 592 nodes and $1,849$ directed edges (self-loops were omitted from the enumeration). The number of edges in the induced undirected graph that occurs after restricting ourselves in these 592 nodes (again, self-loops are omitted) is $1,566$. Table 7.3 gives the distribution of directed shortest path lengths in this directed component as well as the distribution of the undirected shortest path lengths in the undirected graph induced by the concepts that appear in the big directed component induced by the assertions with negative polarity of `ConceptNet 4`.

7.2.2 Positive Polarity

Table 7.4 gives the distribution of the shortest paths in the directed and undirected graph induced by the assertions of the English language with positive score and positive polarity. It also presents the number of pairs for which the second vertex is unreachable from the first one.

Table 7.1: Distribution of shortest paths in the graph induced by the assertions with positive score and negative polarity in `ConceptNet 4`. The table on the left presents the case of the directed graph, while the table on the right presents the case of the undirected graph. The length is equal to ∞ for a pair of vertices when the second vertex is unreachable from the first vertex.

directed graph	
path length	number of shortest paths
1	13,387
2	124,135
3	482,551
4	977,349
5	1,539,103
6	1,461,467
7	1,400,197
8	936,127
9	856,899
10	510,899
11	271,808
12	171,242
13	98,542
14	71,825
15	36,628
16	15,213
17	4,973
18	1,953
19	841
20	424
21	165
22	51
23	9
24	1
∞	78,109,317,723
sum	78,118,293,512

undirected graph	
path length	number of shortest paths
1	12,989
2	8,271,128
3	7,529,595
4	10,133,416
5	6,074,004
6	3,057,701
7	1,191,562
8	400,130
9	121,610
10	57,323
11	37,909
12	34,148
13	14,184
14	6,137
15	1,510
16	366
17	48
18	8
∞	39,022,202,988
sum	39,059,146,756

Positive Polarity: Big Weakly Connected Component

Here we examine the big weakly connected component that arises in the graph induced by the assertions of the English language with positive score and positive polarity. The component has $223,679$ nodes and $383,698$ undirected edges. Table 7.5 gives the distribution of undirected shortest path lengths in the big undirected component induced by the assertions with positive polarity of `ConceptNet 4`.

Positive Polarity: Big Strongly Connected Component

Now we examine the big weakly connected component that arises in the graph induced by the assertions of the English language with positive score and positive polarity. The component has $13,700$ nodes and $120,865$ edges. Table 7.6 gives the distribution of directed shortest path lengths in the big directed component induced by the assertions with positive polarity of `ConceptNet 4` as well as the distribution of the undirected shortest path lengths in the undirected graph induced by the concepts that appear in the big directed component induced by the assertions with positive polarity of `ConceptNet 4`.

Table 7.2: Distribution of undirected shortest path lengths in the big weakly connected component induced by the assertions with negative polarity of `ConceptNet 4`.

path length	number of shortest paths
1	11,247
2	8,270,557
3	7,529,480
4	10,133,389
5	6,074,001
6	3,057,701
7	1,191,562
8	400,130
9	121,610
10	57,323
11	37,909
12	34,148
13	14,184
14	6,137
15	1,510
16	366
17	48
18	8
sum	36,941,310

7.2.3 Both Polarities

Table 7.7 gives the distribution of the shortest paths in the directed and undirected graph induced by the assertions of the English language with positive score and any polarity. It also presents the number of pairs for which the second vertex is unreachable from the first one.

Both Polarities: Big Weakly Connected Component

The big weakly connected component that arises in the graph induced by the assertions of the English language with positive score and no restrictions to polarity has $228,784$ nodes and $394,554$ undirected edges. Table 7.8 gives the distribution of undirected shortest path lengths in the big undirected component induced by the assertions with positive polarity of `ConceptNet 4`.

Both Polarities: Big Strongly Connected Component

The big weakly connected component that arises in the graph induced by the assertions of the English language with positive score and no restrictions to polarity has $14,025$ nodes and $126,151$ edges. Table 7.9 gives the distribution of directed shortest path lengths in the big directed component induced by the assertions with any polarity of `ConceptNet 4` as well as the distribution of the undirected shortest path lengths in the undirected graph induced by the concepts that appear in the big directed component induced by the assertions with any polarity of `ConceptNet 4`.

7.3 Longest Geodesic Paths

Chapter 5 showed that the entire graph is disconnected. Hence, instead of examining the diameter which is formally infinite, we will examine the longest geodesic paths. For the computations we consider subgraphs with positive score on the assertions of the English language.

Table 7.3: Distribution of directed shortest path lengths in the big directed component induced by the assertions with negative polarity of `ConceptNet 4` as well as the distribution of the undirected shortest path lengths in the undirected graph induced by the concepts that appear in the big directed component with negative polarity of `ConceptNet 4`.

directed graph	
path length	number of shortest paths
1	1,849
2	8,458
3	24,779
4	44,834
5	59,644
6	58,813
7	49,665
8	34,593
9	24,926
10	17,389
11	10,916
12	6,382
13	3,589
14	2,260
15	1,010
16	452
17	162
18	74
19	40
20	20
21	12
22	5
sum	349,872

undirected graph	
path length	number of shortest paths
1	1,566
2	24,978
3	62,562
4	56,424
5	23,425
6	5,274
7	655
8	50
9	2
sum	174,936

7.3.1 Negative Polarity

In this section we consider the directed and undirected graph induced by assertions with negative polarity only.

Directed Graph

The longest geodesic path has length 24 and connects the concepts `farmer` (908) and `brass` (27632). The full sequence of the longest geodesic path is given by `farmer` (908) → `farm` (2562) → `zoo` (547) → `country` (640) → `urban` (29003) → `rural` (185019) → `common` (17473) → `occasional` (155305) → `often` (5700) → `never` (126958) → `exist` (2907) → `touch` (5106) → `see` (1161) → `computer` (467) → `human` (80) → `animal` (902) → `man` (7) → `chick` (14872) → `egg` (9339) → `chicken` (191) → `cow` (1613) → `horse` (1412) → `gold` (2266) → `silver` (13722) → `brass` (27632). The justification is given by the following sentences.

1. farmer is not farm
2. farm is not zoo
3. a Zoo is not a kind of country.
4. country is not urban
5. urban is not rural
6. rural is not common
7. common is not occasional
8. occasional is not often
9. often is not never
10. never is not existing

Table 7.4: Distribution of shortest paths in the graph induced by the assertions with positive score and positive polarity in `ConceptNet 4`. The table on the left presents the case of the directed graph, while the table on the right presents the case of the undirected graph. The length is equal to ∞ for a pair of vertices when the second vertex is unreachable from the first vertex.

directed graph	
path length	number of shortest paths
1	412,956
2	20,909,748
3	354,226,806
4	1,867,492,200
5	2,569,306,798
6	988,364,189
7	197,669,166
8	31,493,222
9	4,522,183
10	804,884
11	169,392
12	21,064
13	2,175
14	113
15	5
∞	72,082,898,611
sum	78,118,293,512

undirected graph	
path length	number of shortest paths
1	401,367
2	136,176,653
3	2,601,936,809
4	12,781,641,328
5	8,094,579,408
6	1,203,650,632
7	165,209,595
8	26,997,091
9	4,242,831
10	765,445
11	390,830
12	62,288
13	5,142
14	658
15	104
16	4
∞	14,043,086,571
sum	39,059,146,756

11. Some things that exist you can't touch.
12. touch is not seeing
13. a saw is not a kind of computer.
14. A computer should not want to be a human
15. human is not animal
16. animal is not man
17. men is not chicks
18. chick is not egg
19. egg is not chicken
20. chicken is not cow
21. cow is not horse
22. horses is generally not gold.
23. gold is not silver
24. silver is not brass

Big Strongly Connected Component. The diameter of the big directed component is equal to 22. The full sequence of the diameter is given by `zoo` (547) → `country` (640) → `urban` (29003) → `rural` (185019) → `common` (17473) → `occasional` (155305) → `often` (5700) → `never` (126958) → `exist` (2907) → `touch` (5106) → `see` (1161) → `computer` (467) → `human` (80) → `animal` (902) → `man` (7) → `chick` (14872) → `egg` (9339) → `chicken` (191) → `cow` (1613) → `horse` (1412) → `gold` (2266) → `silver` (13722) → `brass` (27632). The justification is given by the following sentences.

1. a Zoo is not a kind of country.
2. country is not urban
3. urban is not rural
4. rural is not common

Table 7.5: Distribution of undirected shortest path lengths in the big weakly connected component induced by the assertions with positive polarity of `ConceptNet 4`.

path length	number of shortest paths
1	383,698
2	136,170,064
3	2,601,936,595
4	12,781,641,299
5	8,094,579,405
6	1,203,650,632
7	165,209,595
8	26,997,091
9	4,242,831
10	765,445
11	390,830
12	62,288
13	5,142
14	658
15	104
16	4
sum	25,016,035,681

5. common is not occasional
6. occasional is not often
7. often is not never
8. never is not existing
9. Some things that exist you can't touch.
10. touch is not seeing
11. a saw is not a kind of computer.
12. A computer should not want to be a human
13. human is not animal
14. animal is not man
15. men is not chicks
16. chick is not egg
17. egg is not chicken
18. chicken is not cow
19. cow is not horse
20. horses is generally not gold.
21. gold is not silver
22. silver is not brass

The equivalent undirected graph of this component has diameter equal to 9. The full sequence of the diameter in this case is given by `lime` (6416) → `lemon` (14212) → `orange` (15004) → `apple` (4596) → `computer` (467) → `person` (9) → `listen` (75) → `sometimes` (526) → `always` (43553) → `occasional` (155305). The justification is given by the following sentences.

1. lime is not lemon
2. lemon is not orange
3. orange is not apple
4. my computer is not a apple
5. person does not want to be a computer
6. a person doesn't want to listen. / the people don't listen, usually
7. Sometimes we don't listen.
8. always is not sometimes

Table 7.6: Distribution of directed shortest path lengths in the big directed component induced by the assertions with positive polarity of `ConceptNet 4` as well as the distribution of the undirected shortest path lengths in the undirected graph induced by the concepts that appear in the big directed component with positive polarity of `ConceptNet 4`.

directed graph	
path length	number of shortest paths
1	120,865
2	4,006,764
3	35,415,728
4	82,100,213
5	52,346,292
6	11,632,181
7	1,709,055
8	283,094
9	52,735
10	8,503
11	800
12	70
sum	187,676,300

undirected graph	
path length	number of shortest paths
1	109,378
2	8,211,734
3	47,622,303
4	35,784,328
5	2,084,925
6	25,409
7	73
sum	93,838,150

9. occasional is not always

Big Weakly Connected Component. In the following section we will see that the longest geodesic path in the undirected graph induced by the assertions of the English language with positive score and negative polarity is 18. This fact, together with the decomposition of the weakly connected components of the graph that is induced by the assertions of the English language with negative polarity and which is presented in Chapter 5 (Table 5.2), it follows that the diameter of this component is equal to 18. One detailed instance admitting this diameter is given in the following section which describes the longest geodesic path in the graph induced by the assertions with negative polarity.

Undirected Graph

The longest geodesic path has length 18 and connects the concepts `twin` (13665) and `height` (96373). The full sequence of the longest geodesic path is given by `twin` (13665) → `look alike` (58776) → `bell` (10210) → `verb` (1490) → `subject` (6754) → `king` (1443) → `queen` (9693) → `america` (2852) → `monarchy` (18801) → `republic` (46056) → `dictatorship` (22962) → `person` (9) → `late` (1520) → `recent` (52116) → `long` (9087) → `wide` (27291) → `narrowness` (345590) → `width` (130163) → `height` (96373). The justification is given by the following sentences.

1. Twins don't necessarily look alike
2. all bells do not look alike
3. "Bell"is not a verb.
4. subject is not verb
5. The king is not a subject.
6. king is not queen
7. America does not have a queen.
8. America is not a monarchy.
9. republic is not monarchy
10. republic is not dictatorship
11. a person doesn't want dictatorship.
12. person does not want to be late
13. recent is not late
14. recent is not long

Table 7.7: Distribution of shortest paths in the graph induced by the assertions with positive score and any polarity in `ConceptNet 4`. The table on the left presents the case of the directed graph, while the table on the right presents the case of the undirected graph. The length is equal to ∞ for a pair of vertices when the second vertex is unreachable from the first vertex.

directed graph	
path length	number of shortest paths
1	424,525
2	23,978,858
3	406,012,505
4	2,045,811,557
5	2,652,013,614
6	979,044,479
7	192,535,914
8	32,167,500
9	5,342,778
10	1,023,297
11	200,328
12	24,916
13	2,471
14	132
15	5
∞	71,779,710,633
sum	78,118,293,512

undirected graph	
path length	number of shortest paths
1	412,569
2	194,269,357
3	3,140,569,387
4	13,521,818,553
5	7,986,933,052
6	1,141,783,884
7	154,810,521
8	25,254,401
9	3,923,373
10	740,256
11	389,913
12	59,126
13	4,761
14	632
15	104
16	4
∞	12,888,176,863
sum	39,059,146,756

15. wide is not long
16. narrowness is not wide
17. narrowness is not width
18. height is not width

7.3.2 Positive Polarity

In this section we consider the directed and undirected graph induced by assertions with positive polarity only.

Directed Graph

The longest geodesic path has length 15 and connects the concepts `american alphabet` (40903) and `mosque` (177603). The full sequence for the path is given by `american alphabet` (40903) → `twenty six letter` (40904) → `english alphabet` (8492) → `26 letter` (2622) → `english language` (2623) → `confuse` (1871) → `ask question` (8559) → `find information` (8931) → `discover new` (87726) → `tell many person` (427796) → `evangelist` (98420) → `fundamentalist` (176617) → `taliban` (119866) → `islamist` (119867) → `muslim` (8663) → `mosque` (177603). The justification is given by the following sentences.

1. The American alphabet contains twenty six letters.
2. There are twenty six letters in the english alphabet
3. The English alphabet has 26 letters.
4. There are 26 letters in the english language.
5. The English language is sometimes confusing.
6. When you are confused about something you should ask questions.
7. If you want to find information then you should ask questions
8. Something that might happen while finding information is that you discover new things
9. discovering something new would make you want to tell many people about something

Table 7.8: Distribution of undirected shortest path lengths in the big weakly connected component induced by the assertions with any polarity of `ConceptNet 4`.

path length	number of shortest paths
1	$394,554$
2	$194,262,673$
3	$3,140,569,163$
4	$13,521,818,522$
5	$7,986,933,049$
6	$1,141,783,884$
7	$154,810,521$
8	$25,254,401$
9	$3,923,373$
10	$740,256$
11	$389,913$
12	$59,126$
13	$4,761$
14	632
15	104
16	4
sum	$25,016,035,681$

10. telling many people about something is for evangelists.
11. evangelist is a type of fundamentalist
12. You are likely to find fundamentalists in the Taliban.
13. The Taliban are Islamists.
14. an Islamist is a kind of Muslim.
15. You are likely to find Muslims in the mosque.

Remark 6 (Polarity Misclassification)**.** *We note that the longest geodesic path that was originally returned had the concept `eat pork` (20781) as the final node for the path. However, this was purely a result of misclassification in the database, since the sentence associated with the edge admitting the connection was `Muslims can eat anything but pork.`. The assertion that justifies the edge has ID* 177981*, with best frame ID equal to* 30 *which implied the form `{1} can {2}`, which in turn implies positive polarity as expected during our search. However, the actual sentence uses the frame for the opposite polarity.*

After the above observation we searched in the database manually to see if we could replace that particular edge with another one that actually has positive polarity. There are indeed five more sentences with positive polarity and the one with the highest score (2) was chosen and presented above. We further note that among the other four sentences that have positive polarity we encounter `a Muslim can fast during Ramadan` and `muslims can fast for ramadan` which connect the concept `muslim` (8663) with `fast during ramadan` (53518) and `fast ramadan` (65620) respectively.

Big Strongly Connected Component. The diameter of this big directed component is equal to 12. The full sequence of the diameter is given by `sixth day week` (2754) → `Friday` (2755) → `day week` (203694) → `calendar` (1228) → `house` (652) → `person` (9) → `discover new` (87726) → `tell many person` (427796) → `evangelist` (98420) → `fundamentalist` (176617) → `taliban` (119866) → `islamist` (119867) → `muslim` (8663). The justification is given by the following sentences.

1. The sixth day of the week is Friday.
2. Friday is a kind of day of the week.
3. calendar has days of the week
4. You are likely to find Calendar in a house.
5. a house is created by people / house has people
6. a person wants to discover new things

Table 7.9: Distribution of directed shortest path lengths in the big directed component induced by the assertions with any polarity of `ConceptNet 4` as well as the distribution of the undirected shortest path lengths in the undirected graph induced by the concepts that appear in the big directed component with any polarity of `ConceptNet 4`.

directed graph	
path length	number of shortest paths
1	126, 151
2	4, 499, 601
3	39, 414, 540
4	86, 974, 943
5	52, 332, 076
6	11, 279, 229
7	1, 667, 767
8	311, 440
9	68, 835
10	10, 879
11	1, 047
12	92
sum	196, 686, 600

undirected graph	
path length	number of shortest paths
1	114, 294
2	9, 938, 647
3	51, 498, 460
4	34, 859, 851
5	1, 908, 614
6	23, 366
7	68
sum	98, 343, 300

7. discovering something new would make you want to tell many people about something
8. telling many people about something is for evangelists.
9. evangelist is a type of fundamentalist
10. You are likely to find fundamentalists in the Taliban.
11. The Taliban are Islamists.
12. an Islamist is a kind of Muslim.

The equivalent undirected graph of this component has diameter equal to 7. The full sequence of the diameter in this case is given by `tell punishment` (978) → `pass sentence` (297) → `word` (51) → `person` (9) → `office build` (210) → `television studio` (15853) → `helsinki` (3075) → `capital finland` (3074). The justification is given by the following sentences.

1. If you want to pass sentence then you should tell somebody their punishment
2. passing sentence requires words
3. I can word this
4. Somewhere someone can be is the office building
5. the television studio is part of the office building
6. You are likely to find a television studio in Helsinki.
7. helsinki is the capital of finland

Big Weakly Connected Component. In the following section we will see that the longest geodesic path in the undirected graph induced by the assertions of the English language with positive score and positive polarity is 16. This fact, together with the decomposition of the weakly connected components of the graph that is induced by the assertions of the English language with positive score and positive polarity and which is presented in Chapter 5 (Table 5.5 and Figure 5.8), it follows that the diameter of this component is equal to 16. One detailed instance admitting this diameter is given in the following section which describes the longest geodesic path in the graph induced by the assertions with positive score and polarity.

Undirected Graph

The longest geodesic path in this case is 16 and connects the concepts `anti-charm quark` (15922) and `double-breasted de fursac jacket` (328674). The full sequence for the path is given by `anti-charm quark` (15922) → `c c-bar meson` (15620) → `charm quark` (15616) → `charm lambda-plus` (15621) → `down quark` (15659) → `neutron`

$$(15664) \rightarrow \texttt{universe}\ (1639) \begin{array}{l}\nearrow \texttt{something}\ (5) \rightarrow \texttt{dress}\ (562) \searrow \\ \searrow \texttt{god}\ (4277) \rightarrow \texttt{armor}\ (30372) \nearrow\end{array} \texttt{vest}\ (15219) \rightarrow \texttt{waistcoat}\ (15873)$$

→ `single-breasted three-piece suit` (311438) → `single-breasted jacket` (311437) → `single-breasted two-piece suit` (311439) → `man suit pant` (311447) → `double-breasted two-piece de fursac suit` (328675) → `double-breasted de fursac jacket` (328674). The justification is given by the following sentences.

1. an anti-charm quark is part of a c c-bar meson
2. a charm quark is part of a c c-bar meson
3. a charm quark is part of a charmed lambda-plus
4. a down quark is part of a charmed lambda-plus
5. a down quark is part of a neutron
6. Something you find in the universe is neutrons
7. Somewhere something can be is the universe / The universe is created by God.
8. A dress is something worn on the body / armor is a type of god
9. vest is to dress / armor is related to vest
10. vest is a type of waistcoat
11. a waistcoat is part of a single-breasted three-piece suit
12. a single-breasted jacket is part of a single-breasted three-piece suit
13. a single-breasted jacket is part of a single-breasted two-piece suit
14. some men's suit pants is part of a single-breasted two-piece suit
15. some men's suit pants is part of a double-breasted two-piece de fursac suit
16. a double-breasted de fursac jacket is part of a double-breasted two-piece de fursac suit

7.3.3 Both Polarities

Finally, in the case where both polarities are allowed on the induced graphs, the longest geodesic paths are the same as in the case of the graphs induced by assertions of positive polarity only.

Directed Graph

In the case of the directed graph the path that was returned was different only in the final edge, where we encountered the connection `muslim` (8663) → `believe jesus god` (51958), admitted by the sentence `Muslims do not believe that Jesus is god.`.

Big Strongly Connected Component. The diameter of this big directed component is equal to 12. The full sequence of the diameter returned by `igraph` is given by `sixth day week` (2754) → `Friday` (2755) → `day week` (203694) → `bathroom` (1007) → `library` (68) → `learn` (401) → `pride` (14745) → `tell many person` (427796) → `evangelist` (98420) → `fundamentalist` (176617) → `taliban` (119866) → `islamist` (119867) → `muslim` (8663). The justification is given by the following sentences.

1. The sixth day of the week is Friday.
2. Friday is a kind of day of the week.
3. You are not likely to find a day of the week in the bathroom.
4. You are likely to find a bathroom in a library
5. library is for learning.
6. Sometimes learning causes pride.
7. pride would make you want to tell many people about something
8. telling many people about something is for evangelists.
9. evangelist is a type of fundamentalist
10. You are likely to find fundamentalists in the Taliban.
11. The Taliban are Islamists.
12. an Islamist is a kind of Muslim.

In the case of the equivalent undirected graph, the diameter is 7 and is identical to the one found in the big strongly connected component that was found in the graph induced by the assertions with positive polarity only. Please refer to that case for the complete description.

Big Weakly Connected Component. In the following section we will see that the longest geodesic path in the undirected graph induced by the assertions of the English language with positive score and any polarity is 16.

This fact, together with the decomposition of the weakly connected components of the graph that is induced by the assertions of the English language with positive score and positive polarity and which is presented in Chapter 5 (Table 5.8 and Figure 5.8 [1]), it follows that the diameter of this component is equal to 16. One detailed instance admitting this diameter has already been given earlier in the examination of the undirected graph induced by the vertices that appear in the big strongly connected component of `ConceptNet 4` induced by the assertions of the English language with positive score and polarity.

Undirected Graph

In the case of the undirected graph the path that was returned was entirely identical to the case of the undirected graph induced by assertions of positive polarity only.

7.4 Summary

Tables 7.10 and 7.11 give a brief summary of the results related to shortest paths that were presented earlier.

Table 7.10: The average shortest path length of the graphs induced by the assertions of the English language with positive score and various polarities, together with the length of the longest geodesic path in each graph. Recall that the graphs are disconnected, and hence the diameter is infinite in every case. Moreover, the last column indicates whether the length of the longest geodesic path is unique in the respective graph or not.

polarity	directed graph	average shortest path	longest geodesic path	unique
negative	✗	3.863	18	✗
negative	✓	6.737	24	✓
positive	✗	4.330	16	✗
positive	✓	4.811	15	✗
both	✗	4.280	16	✗
both	✓	4.772	15	✗

Table 7.11: The average shortest path length of the big components that arise in the graphs induced by the assertions of the English language with positive score and various polarities, together with the length of the diameter in every case. The last column indicates whether the diameter is unique in the respective component or not.

polarity	connected component	oriented edges	average shortest path	diameter	unique
negative	weakly	✗	3.864	18	✗
negative	strongly	✓	6.428	22	✗
negative	strongly	✗	3.537	9	✗
positive	weakly	✗	4.330	16	✗
positive	strongly	✓	4.205	12	✗
positive	strongly	✗	3.337	7	✗
both	weakly	✗	4.280	16	✗
both	strongly	✓	4.167	12	✗
both	strongly	✗	3.291	7	✗

[1] Even though Figure 5.8 refers to the weakly connected components with positive polarity only, it still suffices for our purposes, since the connected components that could be candidates for giving a possible different longest geodesic path, are the same.

Chapter 8

Cliques

In this section we give an overview of the maximal cliques that we encounter in `ConceptNet 4`. The edges that are retained are those that come from assertions with positive score. In every case we examine the induced undirected graph with no self-loops in order to determine the cliques. Table 8.1 presents the number of cliques found in every case.

Table 8.1: The number of maximal cliques as well as the distribution of the maximal cliques for various frequency ranges and both polarities. All relations are allowed but the scores of the assertions have to be positive. As usual the assertions are those in the English language.

polarity	range for frequency values	number of maximal cliques	number of maximal cliques of size									
			3	4	5	6	7	8	9	10	11	12
negative	{-10}	0	0	0	0	0	0	0	0	0	0	0
	{-10, -9}	0	0	0	0	0	0	0	0	0	0	0
	{-10, -9, -8}	0	0	0	0	0	0	0	0	0	0	0
	{-10, ..., -7}	0	0	0	0	0	0	0	0	0	0	0
	{-10, ..., -6}	0	0	0	0	0	0	0	0	0	0	0
	{-10, ..., -5}	835	779	56	0	0	0	0	0	0	0	0
	{-10, ..., -4}	835	779	56	0	0	0	0	0	0	0	0
	{-10, ..., -3}	835	779	56	0	0	0	0	0	0	0	0
	{-10, ..., -2}	836	780	56	0	0	0	0	0	0	0	0
	{-10, ..., -1}	836	780	56	0	0	0	0	0	0	0	0
	{-10, ..., 0}	836	780	56	0	0	0	0	0	0	0	0
positive	{0, ..., 10}	107,100	47,026	28,655	17,884	9,046	3,083	955	314	113	23	1
	{1, ..., 10}	107,100	47,026	28,655	17,884	9,046	3,083	955	314	113	23	1
	{2, ..., 10}	107,100	47,026	28,655	17,884	9,046	3,083	955	314	113	23	1
	{3, ..., 10}	107,097	47,024	28,655	17,883	9,046	3,083	955	314	113	23	1
	{4, ..., 10}	107,097	47,024	28,655	17,883	9,046	3,083	955	314	113	23	1
	{5, ..., 10}	103,946	45,997	27,805	17,181	8,620	2,956	948	305	112	21	1
	{6, ..., 10}	15	15	0	0	0	0	0	0	0	0	0
	{7, ..., 10}	15	15	0	0	0	0	0	0	0	0	0
	{8, 9, 10}	8	8	0	0	0	0	0	0	0	0	0
	{9, 10}	0	0	0	0	0	0	0	0	0	0	0
	{10}	0	0	0	0	0	0	0	0	0	0	0

8.1 Maximum Clique: All Relations, Positive Polarity

There is a unique maximum clique when all relations are allowed in the induced graph of the English assertions with positive score. The largest maximal clique has size 12 and relates the concepts `person`, `build`, `house`, `home`, `apartment`, `room`, `live room`, `couch`, `table`, `chair`, `cat`, and `dog`. The interpretation (*surface form*) of `live room` should be `living room`, or `in a living room`, etc., `build` should be interpreted as `a building`, etc.

8.2 On the Maximal Cliques with Negative Polarity

The clique (triangle) that is introduced when the range for the frequency values is expanded from $\{-10, \ldots, -3\}$ to $\{-10, \ldots, -2\}$ (see Table 8.1) is composed of the concepts `person` (9), `chore` (22621), and `fun` (134); inside the parentheses we can read the IDs of the specific concepts. The justification comes from the sentences `a person doesn't want to do chores.`, `People do things that are not fun.`, and `chores are rarely fun.`. Regarding the maximal cliques of size 4, below we give the list with all 56 of them. Again, inside the parentheses we can read the ID of each concept.

```
scarce (196339), lot (4905), many (6989), much (7917)
second (130981), year (2709), hour (2762), minute (2764)
average (76629), bad (2226), good (2666), best (20709)
where (54686), who (23034), why (5469), when (38265)
middle (52077), start (44963), begin (3695), end (10507)
middle (52077), side (17836), top (7514), bottom (5887)
middle (52077), side (17836), front (2423), back (15583)
far (37745), near (25285), here (6352), away (29340)
sight (18526), smell (1172), taste (14093), touch (5106)
sight (18526), smell (1172), taste (14093), sound (2660)
even (15946), night (8677), morning (15749), afternoon (15914)
even (15946), night (8677), morning (15749), day (2759)
taste (14093), smell (1172), hear (9269), touch (5106)
few (8145), lot (4905), many (6989), much (7917)
many (6989), much (7917), lot (4905), little (8268)
spring (5537), winter (1431), summer (1437), fall (9975)
touch (5106), see (1161), smell (1172), hear (9269)
blue (4808), red (2614), yellow (2616), green (2637)
woman (895), man (7), girl (876), boy (5976)
plant (716), human (80), animal (902), god (4277)
plant (716), human (80), animal (902), die (1227)
person (9), plural (28735), child (178), eye (1160)
person (9), slave (27415), pay (1473), free (19126)
person (9), deaf (23417), listen (75), hear (9269)
person (9), best (20709), bad (2226), good (2666)
person (9), female (15676), man (7), boy (5976)
person (9), program language (13345), computer (467), hot (1438)
person (9), know (13183), right (6079), wrong (2664)
person (9), know (13183), understand (1858), unknown (5613)
person (9), write paper (8025), computer (467), telephone (1790)
person (9), boy (5976), man (7), girl (876)
person (9), wait (2858), money (1240), long hair (5916)
person (9), wallet (2466), money (1240), long hair (5916)
person (9), crime (1803), sleep (425), lie (1395)
person (9), telephone (1790), computer (467), television (1298)
person (9), kill (1466), live (580), die (1227)
person (9), hot (1438), computer (467), television (1298)
person (9), lie (1395), talk (394), dog (537)
person (9), television (1298), computer (467), book (2033)
person (9), clean (344), dirty (170), gerbil (14223)
person (9), clean (344), dirty (170), time (2494)
person (9), bed (156), examination (121), conscious (23506)
person (9), bed (156), examination (121), money (1240)
person (9), examination (121), long hair (5916), money (1240)
person (9), human (80), like play (203698), animal (902)
person (9), human (80), face (8835), money (1240)
person (9), human (80), long hair (5916), money (1240)
person (9), human (80), god (4277), animal (902)
person (9), human (80), die (1227), animal (902)
person (9), human (80), animal (902), fly (9215)
person (9), human (80), computer (467), conscious (23506)
person (9), human (80), computer (467), fly (9215)
person (9), human (80), computer (467), book (2033)
person (9), human (80), computer (467), house (652)
person (9), man (7), animal (902), fly (9215)
person (9), man (7), animal (902), god (4277)
```

8.3 On the Maximal Cliques with Positive Polarity

Table 8.2 presents the maximal cliques in the case of positive polarity and high frequency. In this table, the frequency values are in the set $\{7, 8, 9, 10\}$. More importantly, the first 8 cliques presented in Table 8.2 are maximal cliques from assertions with very high frequency values; i.e. the values for the frequencies are in the set $\{8, 9, 10\}$.

Table 8.2: Concepts participating in maximal cliques with positive polarity and high frequency (the values of the frequency are in the range $\{7, \ldots, 10\}$). The cliques are obtained from assertions in the English language with positive score. Cliques 1-8 are obtained when the frequency values range in $\{8, 9, 10\}$, while cliques 9-15 are obtained when the frequency values range in $\{7, \ldots, 10\}$.

concept			clique															✓
#	id	description	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
1	9	person							✓	✓			✓	✓	✓	✓	✓	7
2	33	tree										✓						1
3	80	human							✓	✓			✓	✓	✓	✓	✓	7
4	137	it									✓							1
5	716	plant						✓										1
6	1114	national highway	✓	✓	✓													3
7	1443	king				✓												1
8	1577	window					✓											1
9	1776	glass					✓											1
10	2637	green						✓				✓						2
11	3571	leave						✓				✓						2
12	3663	heavy									✓							1
13	6491	metal									✓							1
14	8689	wear clothe								✓								1
15	9693	queen				✓												1
16	18322	write right hand							✓									1
17	18735	eat together															✓	1
18	20788	avoid pain														✓		1
19	21317	eat table													✓			1
20	21364	live apartment												✓				1
21	22671	clear					✓											1
22	41958	live castle				✓												1
23	69743	federal highway	✓	✓	✓													3
24	69746	well maintain			✓													1
25	69747	wide smooth		✓														1
26	69750	top concrete	✓															1
27	81916	disagree other											✓					1

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
clique size	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
frequency range	8–10								7–10						

Table 8.3 presents the largest and second largest maximal cliques in the case of positive polarity but with moderate frequency. In this table, the frequency values are in the set $\{4, \ldots, 10\}$. Recall that the largest maximal clique is composed of the 12 concepts `person`, `apartment`, `home`, `house`, `build`, `room`, `live room`, `cat`, `couch`, `table`, `dog`, and `chair`. This is the first clique presented in Table 8.3. Figure 8.1 presents the graph induced by the concepts that appear in Table 8.3.

8.4 Maximal Cliques: `ConceptuallyRelatedTo` Relation

We restrict our focus on subgraphs composed of edges with positive score only. In the entire graph, edges representing the relation `ConceptuallyRelatedTo` have positive polarity only. The number of multi-edges is

Table 8.3: Concepts participating in maximal cliques with positive polarity and moderate frequency (the values of the frequency are in the range $\{4, \ldots, 10\}$). The cliques are obtained from assertions in the English language with positive score. Moreover, cliques 23 and 24 are obtained when the frequency ranges in $\{4, \ldots, 10\}$, while all the other cases can also be obtained when the frequency ranges in $\{5, \ldots, 10\}$ as well.

concept			clique																								✓
#	id	description	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	
1	5	something											✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	13
2	7	man																						✓			1
3	9	person	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	24
4	21	town				✓	✓	✓	✓	✓												✓	✓		✓	✓	9
5	67	love																						✓			1
6	68	library												✓			✓		✓		✓	✓	✓				6
7	73	school												✓			✓		✓		✓	✓	✓				6
8	80	human					✓		✓	✓														✓			4
9	156	bed		✓											✓									✓			3
10	178	child																						✓			1
11	467	computer																✓	✓				✓				3
12	537	dog	✓		✓	✓	✓	✓	✓	✓		✓															8
13	580	live								✓																	1
14	596	chair	✓		✓	✓	✓	✓	✓		✓	✓				✓	✓			✓	✓	✓			✓	✓	15
15	616	cat	✓	✓	✓	✓		✓			✓	✓															7
16	652	house	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓				21
17	688	hotel																								✓	1
18	876	girl																						✓			1
19	895	woman																						✓			1
20	1013	city						✓	✓	✓												✓	✓		✓	✓	7
21	1043	desk									✓	✓	✓	✓		✓	✓	✓	✓	✓	✓						10
22	1045	home	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓		✓		✓				✓			16
23	1072	couch	✓	✓																							2
24	1104	build	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	23
25	1111	restaurant																							✓	✓	2
26	1372	bedroom			✓	✓	✓				✓	✓	✓		✓	✓		✓		✓				✓			11
27	1414	store																							✓		1
28	2033	book											✓	✓	✓			✓	✓				✓				6
29	2480	room	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	23
30	2825	sex																						✓			1
31	4570	place																		✓	✓	✓			✓	✓	5
32	5558	bar																							✓	✓	2
33	5581	live room	✓	✓																							2
34	5665	table	✓	✓	✓						✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓						14
35	6062	floor									✓		✓	✓	✓	✓	✓										6
36	19557	apartment	✓	✓	✓	✓	✓	✓	✓	✓																	8

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
clique size	12	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11
frequency range	5–10																						4–10	

23010 and this is the same as the number of directed edges in the induced directed graph. The number of undirected edges is 21848. If we neglect the self-loops the number of multi-edges is 22989 and this is the same as the number of directed edges in the induced directed graph. The number of undirected edges is 21827.

There are 2,199 maximal cliques of size 3, 364 maximal cliques of size 4, 61 maximal cliques of size 5, and 3 maximal (maximum) cliques of size 6. The 3 maximum cliques (that is of size 6) are among the concepts:

1. `circle` (14472), `round` (7057), `ball` (263), `sphere` (5508), `eye` (1160), and `head` (10228).
2. `circle` (14472), `round` (7057), `ball` (263), `sphere` (5508), `eye` (1160), and `egg` (9339).
3. `person` (9), `man` (7), `female` (15676), `girl` (876), `woman` (895), and `doll` (1931).

Cliques of size 5 which are related with the above are the following.

1. `circle` (14472), `round` (7057), `ball` (263), `sphere` (5508), and `drop` (1846).
2. `person` (9), `sister` (3656), `mother` (301), `girl` (876), `female` (15676)
3. `person` (9), `family` (915), `mother` (301), `dad` (9672), `father` (13663)
4. `person` (9), `mother` (301), `girl` (876), `woman` (895), `female` (15676)
5. `person` (9), `man` (7), `father` (13663), `male` (6169), `dad` (9672)
6. `person` (9), `man` (7), `statue` (6436), `woman` (895), `doll` (1931)
7. `person` (9), `man` (7), `male` (6169), `brother` (2383), `boy` (5976)

The rest of the cliques of size 5 are given in raw format below.

```
fluffy white (339045), cloud (446), sheep (6424), wool (6425), cotton (9729)
```

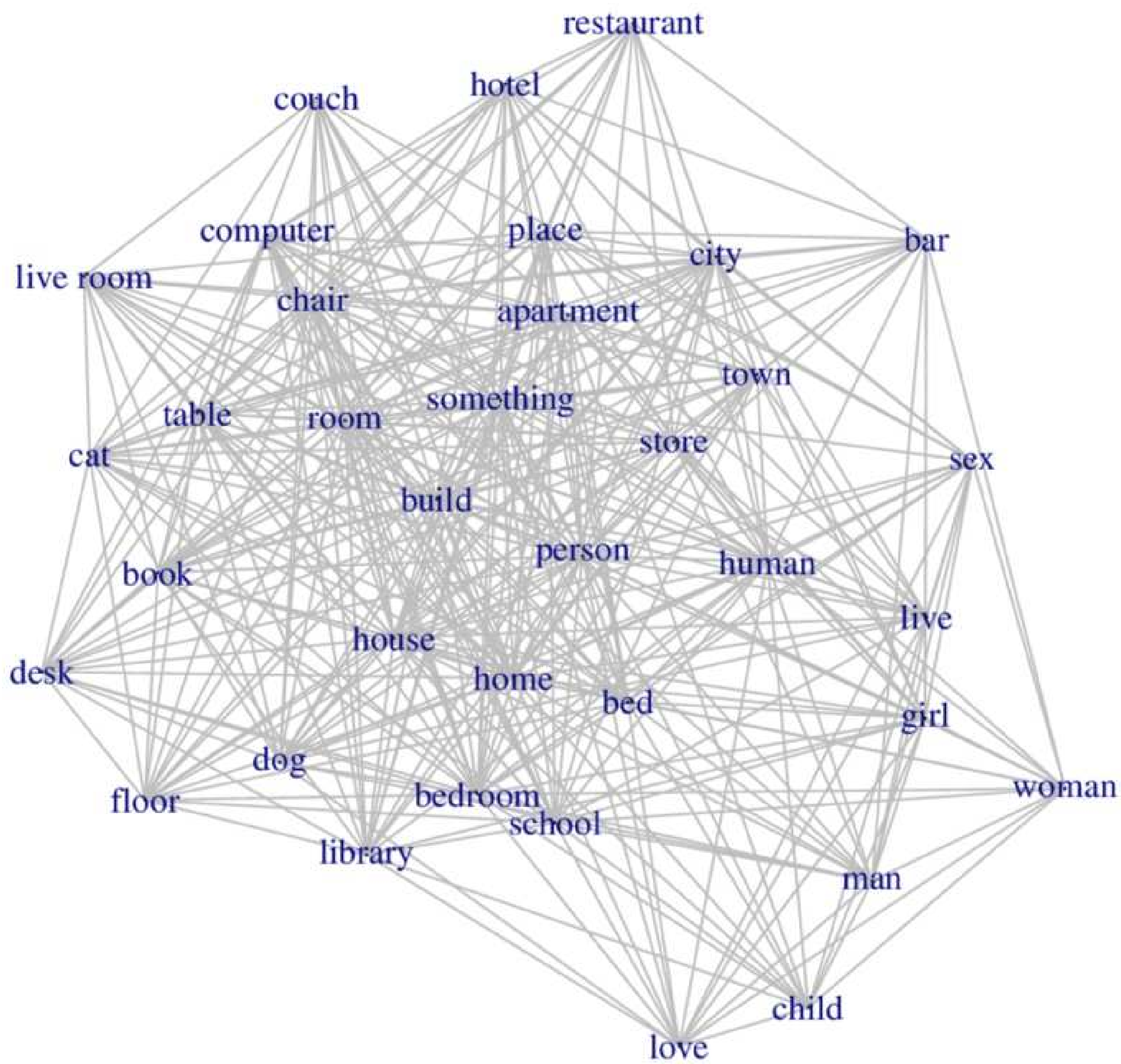


Figure 8.1: The subgraph induced by the concepts that appear in Table 8.3.

```
ground (184976), land (8060), soil (13912), earth (1633), dirt (15359)
cost (81860), price (14042), buy (475), purchase (18262), payment (25417)
cost (81860), price (14042), buy (475), purchase (18262), pay (1473)
cost (81860), price (14042), buy (475), money (1240), payment (25417)
cost (81860), bill (1245), buy (475), money (1240), payment (25417)
sibling (53730), family (915), brother (2383), sister (3656), daughter (13446)
rectangle (41018), square (4138), paper (149), book (2033), card (13442)
rectangle (41018), square (4138), paper (149), book (2033), page (6264)
fog (32237), smoke (188), mist (16981), cloud (446), steam (17055)
white fluffy (23851), cotton (9729), cloud (446), sheep (6424), wool (6425)
silk (22088), wool (6425), cotton (9729), material (591), fabric (1913)
silk (22088), wool (6425), cotton (9729), material (591), cloth (1903)
fluffy (22025), cotton (9729), wool (6425), cloud (446), sheep (6424)
print (21683), write (1893), paper (149), book (2033), text (4472)
stage (21403), play (372), theatre (4095), act (7272), scene (15813)
injury (18717), hurt (686), pain (1813), wind (2284), cut (6250)
sight (18526), vision (7204), see (1161), look (8821), view (5574)
sight (18526), vision (7204), see (1161), look (8821), eye (1160)
purchase (18262), buy (475), sell (649), sale (13614), price (14042)
purchase (18262), buy (475), sell (649), sale (13614), trade (10511)
steam (17055), smoke (188), cloud (446), white (2612), mist (16981)
grey (15391), bullet (13342), steel (3907), metal (6491), silver (13722)
```

```
son (15379), parent (696), mother (301), daughter (13446), father (13663)
son (15379), parent (696), mother (301), daughter (13446), child (178)
son (15379), parent (696), mother (301), dad (9672), father (13663)
silver (13722), metal (6491), tin (8891), steel (3907), iron (2587)
silver (13722), metal (6491), tin (8891), steel (3907), shiny (1382)
daughter (13446), mother (301), female (15676), girl (876), sister (3656)
daughter (13446), mother (301), father (13663), parent (696), family (915)
daughter (13446), mother (301), child (178), parent (696), family (915)
sign (10388), car (529), street (350), drive (1545), road (2368)
wash (10170), bath (70), shower (173), water (1016), soap (3536)
cotton (9729), wool (6425), sheep (6424), cloud (446), white (2612)
hear (9269), music (542), sound (2660), ear (8314), noise (5363)
hear (9269), music (542), sound (2660), ear (8314), listen (75)
globe (9265), sphere (5508), round (7057), ball (263), eye (1160)
ear (8314), head (10228), eye (1160), nose (1171), face (8835)
ship (8013), boat (2389), sail (385), sea (1347), captain (23817)
draw (7764), art (7424), paint (1338), picture (2360), color (2611)
page (6264), paper (149), book (2033), read (1456), write (1893)
door (6022), window (1577), room (2480), house (652), wall (4030)
furniture (5668), wood (370), chair (596), desk (1043), table (5665)
river (4784), water (1016), sea (1347), ocean (1349), blue (4808)
river (4784), water (1016), sea (1347), ocean (1349), lake (660)
text (4472), paper (149), read (1456), write (1893), book (2033)
boat (2389), sea (1347), ocean (1349), fish (655), water (1016)
hand (2300), body (1861), arm (79), leg (1252), foot (1485)
thunder (2274), sky (1354), cloud (446), weather (1248), rain (1856)
ocean (1349), water (1016), sea (1347), fish (655), lake (660)
lady (1281), woman (895), mother (301), girl (876), female (15676)
lady (1281), woman (895), man (7), girl (876), female (15676)
education (1122), school (73), class (93), learn (401), student (886)
parent (696), family (915), dad (9672), mother (301), father (13663)
```

8.5 Maximal Cliques: `IsA` Relation

We restrict our focus on subgraphs composed of edges with positive score only. We distinguish two cases; negative and positive polarity.

Negative Polarity. The induced directed multigraph and directed graph is composed of 3874 edges, while the number of undirected edges is 3498. Note that self-loops are not taken into account since these do not affect the number of cliques. There are 263 cliques with negative polarity. Out of those, 242 are of size 3, while 21 are of size 4. In particular, the maximal cliques of size 4 are given below.

```
scarce (196339), lot (4905), many (6989), much (7917)
second (130981), year (2709), hour (2762), minute (2764)
average (76629), bad (2226), good (2666), best (20709)
where (54686), who (23034), why (5469), when (38265)
middle (52077), start (44963), begin (3695), end (10507)
middle (52077), side (17836), top (7514), bottom (5887)
middle (52077), side (17836), front (2423), back (15583)
far (37745), near (25285), here (6352), away (29340)
sight (18526), smell (1172), taste (14093), touch (5106)
sight (18526), smell (1172), taste (14093), sound (2660)
even (15946), night (8677), morning (15749), afternoon (15914)
even (15946), night (8677), morning (15749), day (2759)
taste (14093), smell (1172), hear (9269), touch (5106)
fall (9975), winter (1431), summer (1437), spring (5537)
hear (9269), see (1161), smell (1172), touch (5106)
little (8268), lot (4905), many (6989), much (7917)
few (8145), lot (4905), many (6989), much (7917)
blue (4808), red (2614), yellow (2616), green (2637)
woman (895), man (7), girl (876), boy (5976)
person (9), man (7), boy (5976), female (15676)
person (9), man (7), boy (5976), girl (876)
```

Positive Polarity. The induced directed multigraph is composed of 90779 edges, the induced directed graph is composed of 90732 edges, while the number of undirected edges is 88654. Note that self-loops are not taken into account since these do not affect the number of cliques. There are 10132 maximal cliques with positive polarity.

Out of those, 7698 are of size 3, 2033 are of size 4, 359 are of size 5, 41 are of size 6, and 1 clique is of size 7. The maximum clique (that is of size 7) is `relation` (201900), `person` (9), `relative` (8531), `family` (915), `brother` (2383), `sister` (3656), `daughter` (13446). The maximal cliques of size 6 related to the above maximum clique are given below.

1. `relation` (201900), `person` (9), `relative` (8531), `family` (915), `father` (13663), `dad` (9672)
2. `relation` (201900), `person` (9), `relative` (8531), `family` (915), `father` (13663), `mother` (301)

The rest of the maximal cliques of size 6 with positive polarity are listed below.

```
abide (222135), house (652), home (1045), nest (1332), dwell (45162), live place (169946)
ground (184976), land (8060), place (4570), farm (2562), garden (1660), field (8720)
chief (180633), person (9), leader (3561), ruler (4313), king (1443), president (7061)
occasion (126340), birthday (6705), christmas (4290), party (307), event (8862), celebration (29221)
cost (81860), fee (36815), bill (1245), charge (4811), tax (9547), payment (25417)
cost (81860), price (14042), payment (25417), bill (1245), charge (4811), expense (25151)
cost (81860), tax (9547), bill (1245), payment (25417), charge (4811), expense (25151)
twist (71331), shake (5439), dance (1667), move (8737), verb (1490), action (9908)
twist (71331), turn (583), roll (6734), move (8737), verb (1490), action (9908)
twist (71331), turn (583), dance (1667), move (8737), verb (1490), action (9908)
structure (54435), build (1104), house (652), nest (1332), home (1045), dwell (45162)
structure (54435), build (1104), house (652), castle (996), home (1045), dwell (45162)
dwell (45162), house (652), home (1045), build (1104), mansion (25687), castle (996)
possession (40599), own (19972), mine (1210), belong (4322), owner (20525), property (21705)
youth (39134), child (178), person (9), girl (876), boy (5976), young person (14434)
fog (32237), mist (16981), smoke (188), cloud (446), air (6408), steam (17055)
poison (24474), water (1016), drink (120), wine (7522), beverage (10164), liquid (1551)
poison (24474), water (1016), drink (120), wine (7522), beverage (10164), food (1359)
property (21705), noun (7478), place (4570), farm (2562), house (652), home (1045)
primate (20931), animal (902), mammal (4850), human (80), man (7), person (9)
gender (19976), person (9), female (15676), girl (876), woman (895), daughter (13446)
rat (19911), animal (902), mammal (4850), rodent (6841), mouse (1284), hamster (15121)
rat (19911), animal (902), mammal (4850), rodent (6841), mouse (1284), squirrel (6609)
female (15676), woman (895), person (9), sister (3656), girl (876), daughter (13446)
female (15676), woman (895), person (9), human (80), girl (876), chick (14872)
female (15676), woman (895), person (9), human (80), girl (876), daughter (13446)
female (15676), woman (895), person (9), human (80), girl (876), lady (1281)
female (15676), woman (895), person (9), human (80), girl (876), mother (301)
son (15379), person (9), brother (2383), family (915), relative (8531), daughter (13446)
son (15379), person (9), child (178), family (915), daughter (13446), relative (8531)
son (15379), person (9), child (178), family (915), daughter (13446), kid (5854)
young person (14434), person (9), child (178), kid (5854), girl (876), boy (5976)
action (9908), activity (6207), exercise (61), move (8737), walk (97), run (1102)
action (9908), verb (1490), move (8737), pass (9934), drive (1545), go (2801)
cotton (9729), fabric (1913), wool (6425), material (591), cloth (1903), textile (8844)
cotton (9729), fabric (1913), wool (6425), material (591), cloth (1903), linen (1137)
voice (8828), sound (2660), call (1061), talk (394), speak (1305), communication (6769)
field (8720), place (4570), farm (2562), garden (1660), area (1915), land (8060)
this (4539), it (137), live room (5581), house (652), home (1045), place (4570)
```

8.6 Maximal Cliques: `UsedFor` Relation

We restrict our focus on subgraphs composed of edges with positive score only. We distinguish two cases; negative and positive polarity.

Negative Polarity. The induced directed multigraph, directed graph, and undirected graph is composed of 193 edges in each case. Again, self-loops are not taken into account since these do not affect the number of cliques. There is only maximal clique in this case and it has size 3. It is composed of the concepts `gerbil` (14223), `exercise` (61), `drive car` (1005).

Positive Polarity. The induced directed multigraph as well as the induced directed graph is composed of 50228 edges, while the number of undirected edges is 50016. Again, self-loops are not taken into account since these do not affect the number of cliques. There are 4427 maximal cliques with positive polarity. Out of those, 3667 are of size 3, 686 are of size 4, 73 are of size 5, and 1 is of size 6. The maximum clique (that is of size 6) is `get`

drunk (310177), fun (134), party (307), drink alcohol (1386), buy beer (5734), and celebrate (13996). The maximal cliques of size 5 related to the above maximum clique are given below.

1. get drunk (310177), fun (134), go party (24657), buy beer (5734), celebrate (13996)
2. get drunk (310177), fun (134), party (307), socialis (29314), nightclub (10669)

The rest of the maximal cliques of size 5 with positive polarity are listed below.

```
get clean (315026), bath (70), take bath (1316), soap (3536), hygiene (13991)
get shape (312438), improve health (10456), go jog (260), jog (6511), lose weight (10298)
get shape (312438), lose weight (10298), jog (6511), health (9745), go run (423)
get shape (312438), lose weight (10298), jog (6511), health (9745), go jog (260)
get physical activity (312389), get exercise (311524), fun (134), enjoyment (643), play lacrosse (28752)
meet person (119411), socialize (27285), party (307), dance (1667), dance club (21969)
meet person (119411), network (6746), socialis (29314), make friend (71547), hang out bar (427)
meet person (119411), network (6746), socialis (29314), make friend (71547), party (307)
meet person (119411), fun (134), socialis (29314), hang out bar (427), make friend (71547)
meet person (119411), fun (134), socialis (29314), party (307), make friend (71547)
meet person (119411), fun (134), socialis (29314), party (307), nightclub (10669)
meet person (119411), fun (134), dance club (21969), party (307), dance (1667)
meet person (119411), fun (134), dance (1667), party (307), nightclub (10669)
remove dirt (42600), soap (3536), clean (344), wash (10170), bathe (26690)
remove dirt (42600), soap (3536), clean (344), wash (10170), bath (70)
see band (25769), dance (1667), fun (134), enjoyment (643), listen music (642)
see band (25769), dance (1667), fun (134), enjoyment (643), music (542)
see band (25769), dance (1667), fun (134), enjoyment (643), hear music (45)
go party (24657), fun (134), celebrate (13996), buy beer (5734), good time (7209)
dance club (21969), dance (1667), listen music (642), fun (134), party (307)
give information (18633), talk (394), make phone call (402), call (1061), telephone (1790)
classroom (18421), learn (401), class (93), student (886), teach (1052)
purchase (18262), buy (475), store (1414), sale (13614), price (14042)
celebrate (13996), pub (13545), party (307), drink alcohol (1386), buy beer (5734)
celebrate (13996), fun (134), good time (7209), party (307), buy beer (5734)
nightclub (10669), fun (134), party (307), listen music (642), dance (1667)
watch television (10343), relax (4187), rest (310), sit down (3442), beanbag chair (9797)
watch television (10343), relax (4187), rest (310), sleep (425), beanbag chair (9797)
watch television (10343), relax (4187), rest (310), sleep (425), sofa (6231)
exchange information (10018), conversation (390), talk (394), telephone (1790), communicate (9747)
business (9787), telephone (1790), conversation (390), talk (394), make phone call (402)
communicate (9747), talk (394), make phone call (402), telephone (1790), call (1061)
communicate (9747), talk (394), make phone call (402), telephone (1790), conversation (390)
egg (9339), chicken (191), eat (432), cook (946), food (1359)
find information (8931), learn (401), research (1978), surf web (203), computer (467)
move (8737), transportation (2364), drive (1545), car (529), highway (2851)
move (8737), travel (1143), highway (2851), car (529), drive (1545)
see movie (7891), go film (305), fun (134), entertain (100), enjoyment (643)
enjoy yourself (7798), fun (134), sex (2825), procreate (4344), copulate (5623)
toy (7701), play (372), entertainment (607), ball (263), game (732)
toy (7701), play (372), fun (134), ball (263), game (732)
communication (6769), talk (394), make phone call (402), telephone (1790), call (1061)
communication (6769), talk (394), make phone call (402), telephone (1790), conversation (390)
play game (6081), entertainment (607), pass time (5077), surf web (203), computer (467)
play game (6081), fun (134), surf web (203), computer (467), pass time (5077)
play game (6081), fun (134), surf web (203), computer (467), learn (401)
song (6068), listen music (642), sing (5711), fun (134), enjoyment (643)
song (6068), music (542), sing (5711), fun (134), enjoyment (643)
song (6068), entertain (100), fun (134), enjoyment (643), sing (5711)
sing (5711), music (542), fun (134), enjoyment (643), guitar (989)
sing (5711), music (542), fun (134), enjoyment (643), tell story (199)
sing (5711), entertain (100), fun (134), tell story (199), enjoyment (643)
copulate (5623), fun (134), sex (2825), pleasure (4231), procreate (4344)
relax (4187), enjoyment (643), dance (1667), party (307), listen music (642)
literature (3901), learn (401), study (122), read (1456), research (1978)
sport (2130), fun (134), ball (263), play (372), game (732)
book (2033), show (2243), entertain (100), fun (134), enjoyment (643)
book (2033), read (1456), learn (401), study (122), text (4472)
book (2033), read (1456), learn (401), study (122), research (1978)
doll (1931), fun (134), child (178), play (372), learn (401)
dance (1667), fun (134), enjoyment (643), party (307), listen music (642)
dance (1667), fun (134), enjoyment (643), party (307), entertain (100)
go sleep (1207), bed (156), dream (172), go bed (406), relaxation (4254)
```

```
go movie (920), fun (134), enjoyment (643), entertain (100), watch movie (265)
student (886), learn (401), school (73), class (93), teach (1052)
computer (467), research (1978), learn (401), study (122), read (1456)
sleep (425), bed (156), dream (172), rest (310), relaxation (4254)
go bed (406), bed (156), relaxation (4254), dream (172), rest (310)
go film (305), fun (134), watch movie (265), entertain (100), enjoyment (643)
tell story (199), fun (134), enjoyment (643), entertain (100), show (2243)
library (68), read (1456), learn (401), study (122), research (1978)
```

8.7 Maximal Cliques: `LocatedNear` Relation

We restrict our focus on subgraphs composed of edges with positive score only. There are only edges with positive polarity. The induced directed multigraph as well as the induced directed graph is composed of 5043 edges, while the number of undirected edges is 4846. Again, self-loops are not taken into account since these do not affect the number of cliques. There are 385 maximal cliques (with positive polarity). Out of those, 358 are of size 3, 23 are of size 4, and 4 are of size 5. The maximum cliques are:

1. `shore` (20212), `sea` (1347), `ocean` (1349), `coast` (6350), `wave` (8813)
2. `wave` (8813), `beach` (24), `sea` (1347), `ocean` (1349), `coast` (6350)
3. `water` (1016), `sea` (1347), `ocean` (1349), `beach` (24), `coast` (6350)
4. `water` (1016), `sea` (1347), `ocean` (1349), `beach` (24), `sand` (5768)

The 23 maximal cliques of size 4 with positive polarity are listed below.

```
ground (184976), floor (6062), foot (1485), bottom (5887)
ground (184976), plant (716), seed (9375), dirt (15359)
crop (33366), farmer (908), farm (2562), field (8720)
stick (31425), tree (33), wood (370), forest (1747)
cheek (15623), nose (1171), eye (1160), face (8835)
soil (13912), plant (716), garden (1660), seed (9375)
head (10228), ear (8314), eye (1160), face (8835)
bear (10208), forest (1747), tree (33), wood (370)
test (9242), school (73), student (886), teacher (3556)
face (8835), eye (1160), nose (1171), ear (8314)
field (8720), farm (2562), horse (1412), barn (4112)
squirrel (6609), tree (33), wood (370), forest (1747)
air (6408), cloud (446), bird (962), sky (1354)
door (6022), house (652), window (1577), room (2480)
table (5665), plate (1604), dinner (1605), napkin (1698)
soap (3536), bath (70), tub (1006), wash (10170)
soap (3536), bath (70), tub (1006), shower (173)
crab (1334), beach (24), sea (1347), ocean (1349)
water (1016), pier (25602), beach (24), ocean (1349)
water (1016), sea (1347), ocean (1349), mist (16981)
water (1016), sea (1347), ocean (1349), blue (4808)
water (1016), sea (1347), ocean (1349), boat (2389)
water (1016), sea (1347), ocean (1349), fish (655)
```

8.8 Maximal Cliques: `SimilarSize` Relation

We restrict our focus on subgraphs composed of edges with positive score only. There are only edges with positive polarity. The induced directed multigraph as well as the induced directed graph is composed of 1509 edges, while the number of undirected edges is 1459. Again, self-loops are not taken into account since these do not affect the number of cliques. There are 55 maximal cliques (with positive polarity) all of which are of size 3 and are listed below.

```
pin head (333090), flea (25677), louse (40958)
footstep (332497), foot (1485), shoe (2790)
person person (311652), human (80), body (1861)
two person (151400), bed (156), mattress (35203)
fist (99732), hand (2300), apple (4596)
handkerchief (63112), napkin (1698), small towel (28420)
cathedral (58941), temple (15854), big build (32344)
dime (52812), penny (1071), cent (14994)
dime (52812), penny (1071), coin (3573)
crumb (47406), salt (1817), flea (25677)
```

```
infinity (40966), everything (1262), space (4435)
branch (37065), twig (14431), stick (31425)
tiny (35891), ant (14190), flea (25677)
tiny (35891), ant (14190), little (8268)
golf ball (35075), eye (1160), egg (9339)
singular (33174), eye (1160), egg (9339)
flea (25677), ant (14190), bug (5563)
flea (25677), sand (5768), grain (14893)
flea (25677), sand (5768), dust (5736)
flea (25677), salt (1817), grain (14893)
rat (19911), squirrel (6609), rodent (6841)
thumb (15862), finger (3399), bullet (13342)
quarter (15172), eye (1160), ring (7720)
olive (15042), eye (1160), grape (1366)
crown (15023), hat (629), head (10228)
skunk (14906), cat (616), squirrel (6609)
grain (14893), seed (9375), sand (5768)
grain (14893), seed (9375), rice (1510)
card (13442), paper (149), envelope (5487)
card (13442), paper (149), book (2033)
head (10228), plate (1604), face (8835)
sock (10193), foot (1485), shoe (2790)
cucumber (9642), corn (3531), banana (6422)
two (9549), one (581), eye (1160)
seed (9375), pebble (6018), pill (569)
egg (9339), ball (263), eye (1160)
face (8835), plate (1604), hand (2300)
atmosphere (8084), sky (1354), air (6408)
rabbit (7815), squirrel (6609), rodent (6841)
rabbit (7815), squirrel (6609), cat (616)
wolf (6387), dog (537), fox (1746)
record (6029), plate (1604), frisbee (2597)
envelope (5487), paper (149), letter (960)
cup (4116), drink (120), glass (1776)
bowl (3463), nest (1332), plate (1604)
hand (2300), nest (1332), plate (1604)
ocean (1349), water (1016), sea (1347)
person (9), slave (27415), servant (13683)
person (9), grave (14465), coffin (14874)
person (9), grave (14465), body (1861)
person (9), sister (3656), brother (2383)
person (9), clothe (2415), body (1861)
person (9), human (80), dummy (72290)
person (9), human (80), coffin (14874)
person (9), human (80), body (1861)
```

8.9 Maximal Cliques: `ReceivesAction` Relation

We restrict our focus on subgraphs composed of edges with positive score only. There are only edges with positive polarity. The induced directed multigraph, the induced directed graph, as well as the induced undirected graph, are composed of 10845 edges. Again, self-loops are not taken into account since these do not affect the number of cliques. There are 23 maximal cliques (with positive polarity) all of which are of size 3. These maximum cliques are listed below.

```
emac (405208), close (2222), open (6539)
spoken (312030), english (7102), language (9326)
eaten (310995), bagel (6956), toast (31357)
project onto screen (153271), movie (213), film (544)
fist (99732), close (2222), open (6539)
build brick (80792), house (652), build (1104)
find mall (75852), store (1414), clothe (2415)
find mailbox (69073), letter (960), mail (4691)
pothole (33862), repair (2289), broken (311852)
find house (33328), carpet (3450), floor (6062)
catch hook (27363), catfish (654), fish (655)
puncture (22394), repair (2289), broken (311852)
tune (19043), play (372), instrument (2086)
keep pet (18614), cat (616), animal (902)
feed (12213), cat (616), animal (902)
```

```
soda (9270), store (1414), open (6539)
open (6539), close (2222), door (6022)
open (6539), close (2222), window (1577)
open (6539), close (2222), door lock (1250)
open (6539), book (2033), store (1414)
person (9), worker (14094), fire (2895)
person (9), daughter (13446), born (3501)
person (9), kill (1466), murder (2663)
```

Part III

Communities

Chapter 9

Non-Overlapping Communities

In this chapter we will go through results based on non-overlapping community finding algorithms. In every case we apply the community-finding algorithms to graphs (no multiple edges) that are undirected and without any self-loops. So far all the algorithms used have been implemented in `igraph` [4, version 0.6.1].

We will use π to indicate the coreness of the vertices. Modularity will be denoted by μ. The number of communities found by an algorithm in a single run will be denoted by κ. Finally, we will denote with $|C|$ the number of connected components for every graph that we are going to examine. The graph will be clear from the context and it will be induced by vertices that have coreness at least a minimum value.

9.1 Negative Polarity

First we will examine the graphs induced by assertions with negative polarity only. Table 9.1 gives an overview of the results achieved by various community-finding algorithms that have been implemented in `igraph`.

Table 9.1: Overall comparison of the community finding algorithms implemented in `igraph`. We use π to indicate coreness. We can see the average number $\overline{\kappa}$ of communities found per run, as well as the average modularity $\overline{\mu}$ achieved by each algorithm. Bold entries in the columns with the modularities indicate the maximum value achieved among all algorithms per row; that is per subgraph induced by vertices that have coreness at least a certain lower bound value. The best values for the modularity are achieved by the spinglass algorithm and the multilevel algorithm. We do not see a result for spinglass in the last row (that is coreness $\geqslant 2$) because the implementation of the algorithm expects a connected graph. Hence we need to run the algorithm in each connected component. However, the difference in terms of actual computation time is huge compared to the multilevel algorithm.

subgraph properties				spin glass		eigen-vectors		walktrap		between-ness		fast greedy		multi-level		label propagation		infoMAP	
$\pi \geqslant$	$\|V\|$	$\|E\|$	$\|C\|$	$\overline{\kappa}$	$\overline{\mu}$	$\overline{\kappa}$	$\overline{\mu}$	$\overline{\kappa}$	$\overline{\mu}$	$\overline{\kappa}$	$\overline{\mu}$	$\overline{\kappa}$	$\overline{\mu}$	$\overline{\kappa}$	$\overline{\mu}$	$\overline{\kappa}$	$\overline{\mu}$	$\overline{\kappa}$	$\overline{\mu}$
6	68	308	1	5.8	0.283	4	0.247	5	0.225	22	0.163	4	0.260	5	0.265	1.1	0.010	1.0	0.000
5	175	819	1	7.9	0.346	7	0.255	12	0.269	61	0.183	6	0.321	7	0.324	1.0	0.002	1.0	0.000
4	347	1447	1	10.0	0.411	8	0.296	25	0.332	55	0.309	9	0.384	9	0.387	2.2	0.027	17.7	0.308
3	820	2710	1	20.0	0.507	17	0.352	38	0.422	62	0.437	13	0.467	12	0.493	10.6	0.074	72.3	0.447
2	1755	4411	3	–	–	24	0.414	136	0.472	91	0.502	31	0.542	24	0.550	20.4	0.076	178.0	0.487
1	11707	12989	1377	–	–	1434	0.669	1947	0.672	–	–	1506	0.747	1435	0.758	1772.0	0.621	1911.7	0.700

	spin glass	eigen-vectors	walktrap	between-ness	fast greedy	multi-level	label propagation	infoMAP
duration/run (secs)	89.0	98.2	13.6	952.0	37.0	0.1	19.2	21.8

9.1.1 Spin Glass

The algorithm used is the one implemented in `igraph` which is based on [10]. Table 9.2 presents the results when we apply the algorithm for subgraphs in which we include vertices with successively lower coreness and allow

negative polarity on the edges only.

Table 9.2: Applying the Spin Glass algorithm for community finding implemented in `igraph` by successively including vertices with lower coreness on the undirected graph induced by the assertions with negative polarity (self-loops are removed). In every row we have the number of vertices and the number of edges of each such subgraph together with the number of components (|C|) that we find in that subgraph. The next three columns present the number of communities found by the algorithm; the average among all runs, the minimum, and the maximum. The next three columns present the modularity achieved by the algorithm due to the cut induced by the communities; the average among all runs, the minimum, and the maximum. The entire computation lasted about 890.25 for 10 runs; that is about 89.03 seconds per run.

coreness	|V|	|E|	|C|	communities found			modularity		
				avg	min	max	avg	min	max
$\geqslant 6$	68	308	1	5.800	5	7	0.283301	0.281013	0.285530
$\geqslant 5$	175	819	1	7.900	6	9	0.345868	0.343487	0.347246
$\geqslant 4$	347	1447	1	10.000	9	11	0.410580	0.406916	0.412461
$\geqslant 3$	820	2710	1	20.000	17	23	0.507353	0.505082	0.510296
$\geqslant 2$	1755	4411	3	–	–	–	–	–	–
$\geqslant 1$	11707	12989	1377	–	–	–	–	–	–

Spin Glass: Communities in the Inner-Most Core

An instance generating 6 communities with modularity equal to 0.283142 is shown below.

Community 1 (size is 5). `exercise`, `fun`, `drive car`, `gasoline`, `fidelity`

Community 2 (size is 12). `library`, `bath`, `park`, `eat`, `office`, `home`, `eye`, `time`, `space`, `program language`, `cash register`, `singular`

Community 3 (size is 12). `talk`, `dog`, `cat`, `fish`, `animal`, `bird`, `die`, `mouse`, `horse`, `read`, `ear`, `fly`

Community 4 (size is 5). `tree`, `walk`, `plant`, `god`, `transportation device`

Community 5 (size is 18). `drink`, `car`, `music`, `desk`, `kitchen`, `television`, `food`, `drive`, `telephone`, `audience`, `boat`, `cabinet`, `table`, `way`, `competitive activity`, `gerbil`, `software`, `speedo`

Community 6 (size is 16). `person`, `human`, `examination`, `bed`, `computer`, `house`, `money`, `hot`, `potato`, `rain`, `book`, `fire`, `long hair`, `metal`, `brain`, `conscious`

9.1.2 Eigenvectors

The algorithm used is the one implemented in `igraph` which is based on [6]. Table 9.3 presents the results when we apply the algorithm for subgraphs in which we include vertices with successively lower coreness and allow negative polarity on the edges only. The computation lasted 196.3 seconds for 2 runs.

Eigenvectors: Communities in the Inner-Most Core

We have the following communities.

Community 1 (size is 18). `person`, `bath`, `examination`, `fun`, `computer`, `dog`, `house`, `home`, `eye`, `hot`, `rain`, `book`, `time`, `space`, `long hair`, `program language`, `brain`, `conscious`

Table 9.3: Applying the leading eigenvector algorithm for community finding implemented in `igraph` by successively including vertices with lower coreness on the undirected graph induced by the assertions with negative polarity (self-loops are removed). In every row we have the number of vertices and the number of edges of each such subgraph together with the number of components (|C|) that we find in that subgraph. The next three columns present the number of communities found by the algorithm; the average among all runs, the minimum, and the maximum. The next three columns present the modularity achieved by the algorithm due to the cut induced by the communities; the average among all runs, the minimum, and the maximum. The entire computation lasted 196.3 seconds for 2 runs; that is about 98.15 seconds per run.

coreness	|V|	|E|	|C|	communities found			modularity		
				avg	min	max	avg	min	max
$\geqslant 6$	68	308	1	4.000	4	4	0.246674	0.246674	0.246674
$\geqslant 5$	175	819	1	7.000	7	7	0.254509	0.254509	0.254509
$\geqslant 4$	347	1447	1	8.000	8	8	0.295650	0.295650	0.295650
$\geqslant 3$	820	2710	1	17.000	17	17	0.352476	0.352476	0.352476
$\geqslant 2$	1755	4411	3	24.000	24	24	0.414328	0.414328	0.414328
$\geqslant 1$	11707	12989	1377	1434.000	1434	1434	0.668898	0.668898	0.668898

Community 2 (size is 17). `library`, `park`, `talk`, `eat`, `cat`, `fish`, `bird`, `drive car`, `office`, `mouse`, `drive`, `competitive activity`, `ear`, `fly`, `gerbil`, `cash register`, `singular`

Community 3 (size is 15). `tree`, `exercise`, `human`, `walk`, `bed`, `plant`, `animal`, `die`, `money`, `horse`, `fire`, `god`, `metal`, `gasoline`, `transportation device`

Community 4 (size is 18). `drink`, `car`, `music`, `desk`, `kitchen`, `television`, `food`, `read`, `potato`, `telephone`, `audience`, `boat`, `cabinet`, `table`, `way`, `software`, `speedo`, `fidelity`

9.1.3 Walktrap

The algorithm used is the one implemented in `igraph` which is based on [8]. Table 9.4 presents the results when we apply the algorithm for subgraphs in which we include vertices with successively lower coreness and allow negative polarity on the edges only. We use random walks of length 5 throughout all the runs.

Table 9.4: Applying the Walktrap algorithm for community finding implemented in `igraph` by successively including vertices with lower coreness on the undirected graph induced by the assertions with negative polarity (self-loops are removed). We use 5 steps for every random walk generated throughout all our runs. In every row we have the number of vertices and the number of edges of each such subgraph together with the number of components (|C|) that we find in that subgraph. The next three columns present the number of communities found by the algorithm; the average among all runs, the minimum, and the maximum. The next three columns present the modularity achieved by the algorithm due to the cut induced by the communities; the average among all runs, the minimum, and the maximum. The entire computation lasted 27.19 seconds for 2 runs.

coreness	|V|	|E|	|C|	communities found			modularity		
				avg	min	max	avg	min	max
$\geqslant 6$	68	308	1	5.000	5	5	0.225064	0.225064	0.225064
$\geqslant 5$	175	819	1	12.000	12	12	0.269032	0.269032	0.269032
$\geqslant 4$	347	1447	1	25.000	25	25	0.331810	0.331810	0.331810
$\geqslant 3$	820	2710	1	38.000	38	38	0.421854	0.421854	0.421854
$\geqslant 2$	1755	4411	3	136.000	136	136	0.471808	0.471808	0.471808
$\geqslant 1$	11707	12989	1377	1947.000	1947	1947	0.671694	0.671694	0.671694

Walktrap: Communities in the Inner-Most Core

We have the following communities.

Community 1 (size is 13). `tree`, `walk`, `examination`, `bed`, `plant`, `die`, `money`, `mouse`, `god`, `long hair`, `brain`, `conscious`, `transportation device`

Community 2 (size is 22). `person`, `exercise`, `bath`, `human`, `fun`, `computer`, `music`, `house`, `drive car`, `eye`, `hot`, `potato`, `telephone`, `rain`, `book`, `time`, `fire`, `cabinet`, `table`, `metal`, `way`, `gasoline`

Community 3 (size is 12). `talk`, `eat`, `dog`, `cat`, `fish`, `animal`, `bird`, `horse`, `read`, `ear`, `fly`, `fidelity`

Community 4 (size is 13). `drink`, `car`, `desk`, `kitchen`, `television`, `food`, `drive`, `audience`, `boat`, `competitive activity`, `gerbil`, `software`, `speedo`

Community 5 (size is 8). `library`, `park`, `office`, `home`, `space`, `program language`, `cash register`, `singular`

9.1.4 Betweenness

The algorithm used is the one implemented in `igraph` which is based on [5]. The idea of the algorithm is described below; it is taken from `igraph` documentation online.

> The idea is that the betweenness of the edges connecting two communities is typically high, as many of the shortest paths between nodes in separate communities go through them. So we gradually remove the edge with highest betweenness from the network, and recalculate edge betweenness after every removal. This way sooner or later the network falls off to two components, then after a while one of these components falls off to two smaller components, etc. until all edges are removed. This is a divisive hierarchical approach, the result is a dendrogram.

The algorithm has complexity $\mathcal{O}\left(|V||E|^2\right)$, as the betweenness calculation requires $\mathcal{O}\left(|V||E|\right)$ time and we do it $|E|-1$ times. Hence, we applied the algorithm only on the subgraph induced by the vertices with maximum coreness [1] .

Table 9.5 presents the results when we apply the algorithm for subgraphs in which we include vertices with successively lower coreness and allow negative polarity on the edges only.

Table 9.5: Applying the Edge Betweenness algorithm for community finding implemented in `igraph` by successively including vertices with lower coreness on the undirected graph induced by the assertions with negative polarity (self-loops are removed). In every row we have the number of vertices and the number of edges of each such subgraph together with the number of components ($|C|$) that we find in that subgraph. The next three columns present the number of communities found by the algorithm; the average among all runs, the minimum, and the maximum. The next three columns present the modularity achieved by the algorithm due to the cut induced by the communities; the average among all runs, the minimum, and the maximum. The entire computation lasted about 951.51 seconds for a single run.

coreness	$\lvert V\rvert$	$\lvert E\rvert$	$\lvert C\rvert$	communities found			modularity		
				avg	min	max	avg	min	max
$\geqslant 6$	68	308	1	22.000	22	22	0.163213	0.163213	0.163213
$\geqslant 5$	175	819	1	61.000	61	61	0.183308	0.183308	0.183308
$\geqslant 4$	347	1447	1	55.000	55	55	0.309215	0.309215	0.309215
$\geqslant 3$	820	2710	1	62.000	62	62	0.436608	0.436608	0.436608
$\geqslant 2$	1755	4411	3	91.000	91	91	0.502412	0.502412	0.502412
$\geqslant 1$	11707	12989	1377	–	–	–	–	–	–

[1] Recall that the subgraph has no self-loops.

Betweenness: Communities in the Inner-Most Core

We have the following communities.

Community 1 (size is 9). `person`, `human`, `plant`, `animal`, `die`, `money`, `potato`, `god`, `long hair`

Community 2 (size is 2). `tree`, `brain`

Community 3 (size is 1). `exercise`

Community 4 (size is 26). `library`, `drink`, `park`, `talk`, `car`, `cat`, `fish`, `bird`, `drive car`, `desk`, `kitchen`, `eye`, `mouse`, `television`, `food`, `drive`, `audience`, `book`, `boat`, `table`, `competitive activity`, `ear`, `fly`, `gerbil`, `software`, `speedo`

Community 5 (size is 1). `bath`

Community 6 (size is 2). `walk`, `fun`

Community 7 (size is 1). `examination`

Community 8 (size is 2). `bed`, `conscious`

Community 9 (size is 1). `eat`

Community 10 (size is 9). `computer`, `telephone`, `time`, `space`, `cabinet`, `metal`, `way`, `gasoline`, `program language`

Community 11 (size is 1). `dog`

Community 12 (size is 1). `music`

Community 13 (size is 2). `house`, `home`

Community 14 (size is 1). `office`

Community 15 (size is 1). `horse`

Community 16 (size is 2). `hot`, `fire`

Community 17 (size is 1). `read`

Community 18 (size is 1). `rain`

Community 19 (size is 1). `cash register`

Community 20 (size is 1). `singular`

Community 21 (size is 1). `transportation device`

Community 22 (size is 1). `fidelity`

9.1.5 Fast Greedy

The algorithm used is the one implemented in `igraph` which is based on [2]. According to igraph version 0.6.1 which was used at the time of the writing, some improvements mentioned in [14] have also been implemented. Table 9.6 presents the results when we apply the algorithm for subgraphs in which we include vertices with successively lower coreness and allow negative polarity on the edges only.

Fast Greedy: Communities in the Inner-Most Core

We have the following communities.

Community 1 (size is 19). `person`, `exercise`, `library`, `fun`, `park`, `house`, `office`, `home`, `eye`, `horse`, `hot`, `rain`, `time`, `fire`, `space`, `program language`, `cash register`, `singular`, `fidelity`

Community 2 (size is 17). `drink`, `eat`, `music`, `desk`, `kitchen`, `television`, `food`, `drive`, `audience`, `book`, `cabinet`, `metal`, `way`, `competitive activity`, `gasoline`, `software`, `speedo`

Community 3 (size is 18). `tree`, `human`, `walk`, `examination`, `bed`, `computer`, `plant`, `die`, `money`, `mouse`, `potato`, `telephone`, `god`, `table`, `long hair`, `brain`, `conscious`, `transportation device`

Community 4 (size is 14). `bath`, `talk`, `car`, `dog`, `cat`, `fish`, `animal`, `bird`, `drive car`, `read`, `boat`, `ear`, `fly`, `gerbil`

Table 9.6: Applying the Fast Greedy algorithm for community finding implemented in `igraph` by successively including vertices with lower coreness on the undirected graph induced by the assertions with negative polarity (self-loops are removed). In every row we have the number of vertices and the number of edges of each such subgraph together with the number of components (|C|) that we find in that subgraph. The next three columns present the number of communities found by the algorithm; the average among all runs, the minimum, and the maximum. The next three columns present the modularity achieved by the algorithm due to the cut induced by the communities; the average among all runs, the minimum, and the maximum. The entire computation lasted about 370.2 seconds for 10 runs; that is about 37.02 seconds per run.

coreness	|V|	|E|	|C|	communities found			modularity		
				avg	min	max	avg	min	max
$\geqslant 6$	68	308	1	4.000	4	4	0.260410	0.260410	0.260410
$\geqslant 5$	175	819	1	6.000	6	6	0.320716	0.320716	0.320716
$\geqslant 4$	347	1447	1	9.000	9	9	0.383727	0.383727	0.383727
$\geqslant 3$	820	2710	1	13.000	13	13	0.467128	0.467128	0.467128
$\geqslant 2$	1755	4411	3	31.000	31	31	0.542274	0.542274	0.542274
$\geqslant 1$	11707	12989	1377	1506.000	1506	1506	0.747473	0.747473	0.747473

9.1.6 Multilevel

The algorithm used is the one implemented in `igraph` which is based on [1]. Table 9.7 presents the results when we apply the algorithm for subgraphs in which we include vertices with successively lower coreness and allow negative polarity on the edges only.

Table 9.7: Applying the Multilevel algorithm for community finding implemented in `igraph` by successively including vertices with lower coreness on the undirected graph induced by the assertions with negative polarity (self-loops are removed). In every row we have the number of vertices and the number of edges of each such subgraph together with the number of components (|C|) that we find in that subgraph. The next three columns present the number of communities found by the algorithm; the average among all runs, the minimum, and the maximum. The next three columns present the modularity achieved by the algorithm due to the cut induced by the communities; the average among all runs, the minimum, and the maximum. The entire computation lasted 12.7 seconds for 100 runs; that is about 0.127 seconds per run.

coreness	|V|	|E|	|C|	communities found			modularity		
				avg	min	max	avg	min	max
$\geqslant 6$	68	308	1	5.000	5	5	0.264953	0.264953	0.264953
$\geqslant 5$	175	819	1	7.000	7	7	0.323902	0.323902	0.323902
$\geqslant 4$	347	1447	1	9.000	9	9	0.386802	0.386802	0.386802
$\geqslant 3$	820	2710	1	12.000	12	12	0.492571	0.492571	0.492571
$\geqslant 2$	1755	4411	3	24.000	24	24	0.550032	0.550032	0.550032
$\geqslant 1$	11707	12989	1377	1435.000	1435	1435	0.758024	0.758024	0.758024

Multilevel: Communities in the Inner-Most Core

We have the following communities.

Community 1 (size is 20). `person`, `tree`, `human`, `examination`, `bed`, `computer`, `house`, `home`, `money`, `hot`, `rain`, `book`, `time`, `fire`, `space`, `long hair`, `metal`, `program language`, `brain`, `conscious`

Community 2 (size is 15). `bath`, `drink`, `talk`, `eat`, `dog`, `cat`, `fish`, `bird`, `mouse`, `horse`, `read`, `competitive activity`, `ear`, `fly`, `fidelity`

Community 3 (size is 19). library, fun, car, music, desk, kitchen, eye, television, food, potato, telephone, audience, boat, cabinet, table, way, software, cash register, speedo

Community 4 (size is 8). exercise, plant, animal, drive car, die, god, gasoline, transportation device

Community 5 (size is 6). walk, park, office, drive, gerbil, singular

9.1.7 Label Propagation

The algorithm used is the one implemented in igraph which is based on [9]. Table 9.8 presents the results when we apply the algorithm for subgraphs in which we include vertices with successively lower coreness and allow negative polarity on the edges only.

Table 9.8: Applying the Label Propagation algorithm for community finding implemented in igraph by successively including vertices with lower coreness on the undirected graph induced by the assertions with negative polarity (self-loops are removed). In every row we have the number of vertices and the number of edges of each such subgraph together with the number of components (|C|) that we find in that subgraph. The next three columns present the number of communities found by the algorithm; the average among all runs, the minimum, and the maximum. The next three columns present the modularity achieved by the algorithm due to the cut induced by the communities; the average among all runs, the minimum, and the maximum. Finally the last two columns present in how many runs the algorithm computed as many communities as we had components in that subgraph. The entire computation lasted 1917.0 seconds for 100 runs; that is about 19.17 seconds per run.

coreness	|V|	|E|	|C|	communities found			modularity			agreement	
				avg	min	max	avg	min	max	Y	N
$\geqslant 6$	68	308	1	1.050	1	2	0.010145	0.000000	0.223894	95	5
$\geqslant 5$	175	819	1	1.010	1	2	0.002370	0.000000	0.236981	99	1
$\geqslant 4$	347	1447	1	2.210	2	5	0.027395	0.010952	0.284524	0	100
$\geqslant 3$	820	2710	1	10.590	7	16	0.074087	0.033087	0.371248	0	100
$\geqslant 2$	1755	4411	3	20.410	15	34	0.075914	0.050672	0.410376	0	100
$\geqslant 1$	11707	12989	1377	1772.040	1691	1861	0.620714	0.526809	0.702329	0	100

Label Propagation: Communities in the Inner-Most Core

An example where Label Propagation identifies two communities in the innermost core is shown below.

Community 1 (size is 42). person, tree, exercise, library, bath, human, walk, examination, fun, bed, park, computer, music, house, plant, drive car, office, home, eye, money, hot, potato, telephone, rain, book, time, fire, god, space, cabinet, table, long hair, metal, way, gasoline, program language, gerbil, software, brain, conscious, singular, transportation device

Community 2 (size is 26). drink, talk, eat, car, dog, cat, fish, animal, bird, desk, kitchen, die, mouse, television, food, horse, read, drive, audience, boat, competitive activity, ear, fly, cash register, speedo, fidelity

9.1.8 InfoMAP

The algorithm used is the one implemented in igraph which is based on [12]; see also [11]. Table 9.9 presents the results when we apply the algorithm for subgraphs in which we include vertices with successively lower coreness and allow edges with negative polarity only.

Table 9.9: Applying the InfoMAP algorithm for community finding implemented in `igraph` by successively including vertices with lower coreness on the undirected graph induced by the assertions with negative polarity (self-loops are removed). In every row we have the number of vertices and the number of edges of each such subgraph together with the number of components ($|C|$) that we find in that subgraph. The next three columns present the number of communities found by the algorithm; the average among all runs, the minimum, and the maximum. The next three columns present the codelength of the partitioning found by the algorithm; the average among all runs, the minimum, and the maximum. The next three columns present the modularity achieved by the algorithm due to the cut induced by the communities; the average among all runs, the minimum, and the maximum. Finally the last two columns present in how many runs the algorithm computed as many communities as we had components in that subgraph. The entire computation lasted 217.97 seconds for 10 runs; that is about 21.80 seconds per run.

$\pi \geqslant$	$\|V\|$	$\|E\|$	$\|C\|$	communities found			codelength			modularity			agreement	
				avg	min	max	avg	min	max	avg	min	max	Y	N
6	68	308	1	1.000	1	1	5.993573	5.993573	5.993573	0.000000	0.000000	0.000000	10	0
5	175	819	1	1.000	1	1	7.279672	7.279672	7.279672	0.000000	0.000000	0.000000	10	0
4	347	1447	1	17.700	10	26	7.970802	7.949532	7.981013	0.307898	0.193377	0.385923	0	10
3	820	2710	1	72.300	70	76	8.268610	8.263414	8.277101	0.447178	0.441671	0.454124	0	10
2	1755	4411	3	178.000	170	183	8.330739	8.328037	8.334383	0.487219	0.482917	0.489891	0	10
1	11707	12989	1377	1911.700	1909	1916	6.828170	6.827590	6.828669	0.699840	0.698775	0.700512	0	10

InfoMAP: Communities in the Inner-Most Core

The entire core is one big community. This is the result in all of our runs.

9.2 Positive Polarity

In this section we will examine the graphs induced by assertions with positive polarity only. Table 9.10 gives an overview of the results achieved by various community-finding algorithms that have been implemented in `igraph`.

9.2.1 Spin Glass

The algorithm used is the one implemented in `igraph` which is based on [10]. Table 9.11 presents the results when we apply the algorithm for subgraphs in which we include vertices with successively lower coreness and allow positive polarity on the edges only.

Spin Glass: Communities in the Inner-Most Core

An instance of 7 communities with modularity 0.313939 is shown below.

Community 1 (size is 1). `power`

Community 2 (size is 224). `rock`, `beach`, `tree`, `monkey`, `weasel`, `pant`, `kitten`, `arm`, `human`, `beaver`, `it`, `smoke`, `chicken`, `state`, `ball`, `fungus`, `park`, `trouble`, `snake`, `wood`, `bridge`, `cloud`, `nothing`, `dog`, `zoo`, `live`, `one`, `cat`, `hat`, `country`, `fish`, `lake`, `baby`, `plant`, `hide`, `animal`, `cold`, `moon`, `pet`, `bird`, `shark`, `water`, `rosebush`, `yard`, `sloth`, `bat`, `lizard`, `beautiful`, `eye`, `nose`, `smell`, `well`, `bill`, `snow`, `weather`, `leg`, `everything`, `mouse`, `hole`, `nature`, `bald eagle`, `nest`, `crab`, `ficus`, `sea`, `anemone`, `ocean`, `sun`, `sky`, `grape`, `horse`, `hot`, `kill`, `foot`, `meadow`, `camp`, `den`, `cow`, `earth`, `garden`, `poop`, `outside`, `frog`, `light`, `fox`, `forest`, `marmot`, `mountain`, `drop`, `bone`, `rain`, `body`, `ferret`, `small dog`, `doll`, `lemur`, `name`, `nice`, `museum`, `black`, `canada`, `bad`, `wind`, `hand`, `pee`, `road`, `boat`, `wild`, `war`, `wet`, `flower`, `small`, `new york`, `farm`, `color`, `red`, `stone`, `green`, `life`, `burn`, `large`, `soft`, `fire`, `finger`, `dangerous`, `marmoset`, `australia`, `leave`, `heavy`, `cuba`, `france`, `italy`, `unite state`, `hill`, `apple tree`, `god`, `space`, `mouth`, `river`, `blue`, `grass`, `mammal`, `lot`, `hair`, `measure`, `utah`, `bug`, `tooth`, `sand`, `dictionary`, `rise`, `not`, `bite`, `dark`, `science`, `world`, `air`, `sheep`, `statue`, `warm`, `big`, `high`, `squirrel`, `mean`, `general`, `heat`, `cool`, `skin`, `art`, `noun`, `hard`, `duck`, `ring`, `wyom`, `thing`, `land`, `kill person`, `little`, `ear`, `alive`,

field, wave, face, lawn, long, shade, fly, bee, bear, head, bullet, degree, gas, brown, dirt, adjective, alaska, michigan, maryland, maine, delaware, kansa, be, steam, pretty, decoration, step, part, bush, countryside, same, grow, sunshine, flea, short, outdoors, stick, singular, find outside, branch, wax, generic, ground, unit

Community 3 (size is 1). out

Community 4 (size is 1). course

Community 5 (size is 40). type, word, movie, music, game, family, concert, call, story, speak, band, audience, write, pool, show, sound, news, club, theatre, theater, noise, view, record, song, company, communication, act, crowd, voice, event, hear, organization, pass, end, group, point, stage, race, many person, general term

Community 6 (size is 234). something, town, soup, library, school, plane, class, drink, paper, bed, dirty, gym, office build, wiener dog, box, object, mother, coffee, candle, street, bus, eat, computer, line, milk, tv, drawer, storage, car, vehicle, bottle, turn, material, chair, market, house, hotel, hospital, bank, girl, church, cook, shop, letter, bathroom, city, desk, office, home, couch, kitchen, build, restaurant, spoon, butter, key, electricity, stand, pen, television, magazine, paint, food, bedroom, store, airport, sugar, grocery store, basket, hold, refrigerator, newspaper, rice, surface, liquid, window, oil, cover, plate, dinner, garage, potato, napkin, salad, glass, cupboard, telephone, salt, motel, meat, bookstore, use, cloth, factory, bottle wine, pencil, wheel, book, instrument, trash, can, a, picture, seat, clothe, dish, train station, mall, wallet, room, cell, bicycle, sink, pocket, white, vegetable, shoe, scale, steak, beer, knife, carpet, bowl, corn, fridge, soap, expensive, coin, number, fruit, map, fork, steel, piano, wall, cup, square, shelf, friend house, airplane, phone, this, place, radio, tool, apple, bag, doctor, cheese, bean, make, flat, plastic, container, bar, live room, toilet, cabinet, table, furniture, lamp, pizza, dust, hall, closet, boy, door, floor, meet, basement, sofa, cut, page, college, metal, open, alcohol, university, roll, clock, round, top, wine, jar, put, toy, draw, edible, rug, pot, change, bread, tin, oven, carry, test, egg, building, business, resturant, wash, sock, bell, sign, pantry, note, card, supermarket, machine, roof, circle, cake, solid, useful, handle, department, side, stapler, classroom, transport, apartment, any large city, comfort, edge, case, board, corner, find house, winery, polish, eaten, neighbor house, usually, generic term

Community 7 (size is 368). man, person, train, work, write program, go concert, hear music, exercise, love, bath, listen, go performance, take walk, walk, entertain, run marathon, wait line, attend lecture, study, go walk, play basketball, fun, bore, wait table, go see film, go work, watch tv show, wake up morning, dream, shower, child, go fish, tell story, surf web, play football, go restaurant, visit museum, study subject, live life, go sport event, go play, sit, play soccer, go jog, take shower, play ball, eat food, watch movie, watch film, stretch, play frisbee, go school, surprise, paint picture, go film, party, rest, listen radio, kiss, remember, housework, clean, lunch, watch tv, attend school, play tennis, comfortable, play, take bus, conversation, talk, take course, learn, plan, think, go run, sleep, hang out bar, plan vacation, go see play, attend class, go swim, ride bike, buy, eat restaurant, stress, boredom, ticket, use television, dress, entertainment, listen music, enjoyment, hurt, student, muscle, woman, go movie, enlightenment, stand line, attend classical concert, death, play sport, eat dinner, effort, drive car, traveling, knowledge, teach, laugh joke, run, read book, education, take note, travel, go store, see, go sleep, tire, attention, die, fall asleep, money, run errand, patience, spend money, cry, pay bill, earn money, take bath, drink water, fatigue, take break, hike, drink alcohol, lie, play chess, friend, anger, read, curiosity, pay, swim, break, verb, drive, use computer, take film, smile, fiddle, we, wrestle, see new, dance, fight, job, smart, play baseball, excite, attend rock concert, hear news, contemplate, pain, understand, stay healthy, research, learn new, sweat, headache, fart, read newspaper, sport, understand better, write story, stop, transportation, fall down, practice, help, lose, close eye, satisfaction, time, answer question, perform, need, everyone, go somewhere, good, play card, go, sex, wait, buy ticket, gain knowledge, interest, feel, sit down, ski, surf, teacher, happiness, exhaustion, sit chair, laugh, relax, waste time, pleasure, relaxation, care, procreate, watch, funny, win, go mall, flirt, pass time, shape, climb, wash hand, go home, love else, drunk, peace, sing, buy beer,

internet, kid, like, date, find, play game, activity, jog, quiet, skill, hobby, birthday, tiredness, drink coffee, read magazine, good time, good health, play hockey, eat ice cream, learn language, dive, go zoo, go internet, cash, important, read child, enjoy yourself, see movie, energy, emotion, clean house, fit, view video, play poker, excitement, move, fly airplane, ride horse, stay bed, look, happy, find information, fear, go vacation, breathe, recreation, enjoy, jump, ride bicycle, health, communicate, make money, become tire, action, fall, lose weight, jump up down, watch television, count, healthy, know, learn subject, joy, stand up, information, read letter, lay, jump rope, celebrate, sadness, bike, watch musician perform, motion, feel better, compete, feel good, accident, stay fit, injury, ride, play piano, learn world, see exhibit, release energy, see art, see excite story, orgasm, trip, laughter, express yourself, discover truth, see favorite show, go party, competition, express information, climb mountain, attend meet, fly kite, examine, meet friend, read news, shock, return work, see band, visit art gallery, earn live, punch, cool off, watch television show, socialize, skate, movement, create art, crossword puzzle, enjoy film, go pub, feel happy, play lacrosse, socialis, away, physical activity, get, make person laugh, make friend, chat friend, meet person, meet interest person, get drunk, friend over, get exercise, get tire, enjoy company friend, play game friend, get physical activity, go opus, get shape, sit quietly, do it, get fit, teach other person, entertain person, see person play game

9.2.2 Eigenvectors

The algorithm used is the one implemented in `igraph` which is based on [6]. Table 9.12 presents the results when we apply the algorithm for subgraphs in which we include vertices with successively lower coreness and allow positive polarity on the edges only. The computation lasted 94.98 seconds for 2 runs.

Eigenvectors: Communities in the Inner-Most Core

We have the following communities.

Community 1 (size is 209). something, town, word, library, school, human, plane, it, paper, bed, dirty, gym, office build, box, object, mother, coffee, candle, clean, street, bus, computer, line, drawer, storage, car, vehicle, bottle, material, chair, hat, market, house, hotel, hospital, bank, girl, church, family, shop, letter, bathroom, city, desk, office, home, couch, kitchen, build, restaurant, key, electricity, stand, pen, bill, magazine, paint, bedroom, store, airport, grocery store, hold, refrigerator, newspaper, surface, window, cover, garage, napkin, light, glass, telephone, motel, bookstore, use, cloth, factory, bottle wine, doll, pencil, wheel, name, book, museum, pool, instrument, trash, picture, road, seat, boat, clothe, train station, war, mall, wallet, room, cell, sink, pocket, large, shoe, scale, beer, knife, carpet, soap, expensive, coin, number, heavy, map, fork, steel, piano, wall, cup, square, shelf, friend house, airplane, phone, this, place, radio, tool, bag, doctor, theater, make, measure, flat, plastic, container, bar, live room, toilet, cabinet, table, furniture, lamp, dust, hall, dictionary, closet, boy, door, floor, basement, sofa, page, company, dark, college, metal, open, alcohol, university, clock, noun, hard, toy, ring, crowd, draw, thing, rug, change, carry, organization, building, business, wash, sock, bell, sign, degree, note, card, supermarket, machine, roof, circle, solid, point, useful, handle, department, side, stapler, classroom, transport, step, apartment, part, stage, any large city, comfort, case, board, corner, singular, find house, winery, polish, wax, neighbor house, usually, unit

Community 2 (size is 160). man, train, work, exercise, take walk, walk, run marathon, go walk, play basketball, fun, wait table, go work, shower, child, smoke, go fish, play football, live life, play soccer, go jog, take shower, play ball, stretch, play frisbee, rest, housework, play tennis, play, take bus, plan, go run, sleep, hang out bar, go swim, ride bike, dress, one, turn, hurt, game, muscle, woman, death, play sport, effort, drive car, traveling, run, travel, go store, tire, die, run errand, earn money, drink water, fatigue, take break, hike, lie, anger, kill, swim, break, verb, drive, wrestle, dance, fight, play baseball, attend rock concert, pain, stay healthy, sweat, sport, stop, transportation, fall down, practice, lose, bicycle, go somewhere, life, go, sex, ski, surf, exhaustion, procreate, win, go mall, shape, climb, go home, play game, activity, jog, skill, tiredness, good health, play hockey, cool, dive, cash, energy, clean house, fit, move, fly airplane, ride horse, fear, go vacation, breathe, recreation, jump, ride bicycle, health, become tire, action, pass, fall, lose weight, jump up down,

count, healthy, stand up, lay, jump rope, bike, motion, feel better, compete, feel good, accident, stay fit, injury, ride, release energy, trip, competition, climb mountain, fly kite, race, shock, return work, punch, cool off, skate, movement, go pub, play lacrosse, away, physical activity, get, get drunk, get exercise, get tire, get physical activity, get shape, do it, get fit

Community 3 (size is 245). rock, beach, tree, monkey, soup, weasel, pant, kitten, arm, beaver, drink, chicken, state, wiener dog, ball, eat food, fungus, park, snake, wood, eat, bridge, cloud, milk, dog, zoo, live, cat, country, fish, lake, plant, hide, animal, cold, moon, pet, cook, bird, shark, water, rosebush, yard, sloth, bat, lizard, spoon, butter, beautiful, eye, nose, smell, well, snow, weather, leg, everything, mouse, hole, nature, bald eagle, nest, crab, ficus, sea, anemone, ocean, sun, sky, food, grape, horse, hot, sugar, basket, foot, rice, liquid, meadow, camp, oil, plate, dinner, den, cow, earth, garden, poop, potato, outside, frog, salad, fox, forest, cupboard, marmot, mountain, salt, drop, bone, meat, rain, body, ferret, small dog, lemur, black, canada, can, a, wind, hand, pee, wild, dish, wet, flower, small, new york, farm, color, white, red, stone, vegetable, green, burn, soft, steak, fire, finger, dangerous, marmoset, bowl, australia, corn, fridge, leave, fruit, cuba, france, italy, unite state, hill, apple tree, god, space, apple, mouth, river, blue, grass, cheese, mammal, bean, lot, hair, utah, wash hand, bug, tooth, pizza, sand, rise, not, cut, bite, world, air, sheep, statue, warm, big, high, squirrel, roll, general, round, heat, skin, top, wine, jar, put, duck, edible, wyom, land, pot, little, ear, alive, bread, field, wave, face, lawn, tin, oven, long, shade, fly, egg, bee, resturant, bear, head, group, pantry, bullet, gas, brown, cake, dirt, adjective, alaska, michigan, maryland, maine, delaware, kansa, be, steam, pretty, decoration, out, bush, course, countryside, power, same, edge, grow, sunshine, flea, short, outdoors, stick, find outside, branch, general term, generic, ground, eaten, generic term

Community 4 (size is 255). person, type, write program, go concert, hear music, love, bath, listen, go performance, class, entertain, wait line, attend lecture, study, bore, go see film, watch tv show, wake up morning, dream, tell story, surf web, movie, go restaurant, visit museum, study subject, go sport event, go play, sit, watch movie, watch film, go school, surprise, paint picture, go film, party, listen radio, kiss, remember, lunch, watch tv, attend school, trouble, comfortable, conversation, talk, take course, learn, think, plan vacation, go see play, attend class, nothing, buy, eat restaurant, tv, stress, boredom, ticket, music, use television, entertainment, listen music, enjoyment, baby, student, go movie, enlightenment, stand line, attend classical concert, eat dinner, concert, knowledge, teach, call, laugh joke, read book, education, take note, see, story, go sleep, attention, fall asleep, money, patience, spend money, cry, pay bill, television, speak, take bath, band, drink alcohol, play chess, friend, read, curiosity, pay, use computer, take film, smile, fiddle, we, see new, job, smart, excite, hear news, contemplate, audience, understand, write, research, learn new, nice, headache, fart, read newspaper, understand better, bad, show, write story, help, close eye, satisfaction, time, answer question, perform, need, everyone, sound, good, play card, wait, buy ticket, gain knowledge, news, interest, feel, sit down, teacher, happiness, sit chair, laugh, club, theatre, relax, waste time, pleasure, relaxation, care, watch, funny, flirt, pass time, noise, view, love else, drunk, peace, sing, buy beer, internet, kid, like, date, record, find, song, meet, science, quiet, hobby, birthday, mean, communication, drink coffee, read magazine, good time, act, eat ice cream, learn language, go zoo, go internet, art, important, read child, enjoy yourself, see movie, kill person, emotion, view video, play poker, excitement, stay bed, look, voice, event, happy, find information, test, enjoy, hear, communicate, make money, watch television, end, know, learn subject, joy, information, read letter, celebrate, sadness, watch musician perform, play piano, learn world, see exhibit, see art, see excite story, orgasm, laughter, express yourself, discover truth, see favorite show, go party, express information, attend meet, examine, meet friend, read news, see band, visit art gallery, earn live, watch television show, socialize, create art, crossword puzzle, enjoy film, feel happy, socialis, many person, make person laugh, make friend, chat friend, meet person, meet interest person, friend over, enjoy company friend, play game friend, go opus, sit quietly, teach other person, entertain person, see person play game

9.2.3 Walktrap

The algorithm used is the one implemented in `igraph` which is based on [8]. Table 9.13 presents the results when we apply the algorithm for subgraphs in which we include vertices with successively lower coreness and allow

positive polarity on the edges only. We use random walks of length 5 throughout all the runs.

Remark 7 (Walktrap Memory Requirements)**.** *The drawback of the implementation is that it requires too much memory in order to run. Applying the algorithm on the subgraph induced by vertices with coreness at least* 3 *required about* 2.1 *GBytes of RAM.*

Walktrap: Communities in the Inner-Most Core

We have the following communities.

Community 1 (size is 292). town, rock, beach, tree, monkey, soup, weasel, kitten, plane, beaver, paper, bed, dirty, chicken, state, office build, wiener dog, box, object, coffee, candle, fungus, park, snake, wood, cloud, milk, drawer, storage, zoo, bottle, material, chair, cat, hat, country, market, house, lake, hotel, plant, bank, animal, church, moon, pet, cook, bird, bathroom, city, shark, water, rosebush, yard, desk, office, home, sloth, bat, couch, kitchen, lizard, build, restaurant, spoon, butter, beautiful, key, well, pen, snow, weather, mouse, magazine, hole, nature, bald eagle, nest, crab, ficus, sea, anemone, ocean, sun, sky, food, grape, bedroom, horse, store, airport, sugar, grocery store, basket, hold, refrigerator, newspaper, rice, surface, liquid, meadow, window, oil, cover, plate, dinner, cow, earth, garage, garden, poop, potato, outside, frog, napkin, light, salad, fox, forest, glass, cupboard, marmot, mountain, salt, motel, bone, meat, bookstore, ferret, small dog, cloth, factory, bottle wine, pencil, lemur, black, canada, trash, can, a, picture, wild, clothe, dish, mall, flower, wallet, room, small, new york, farm, sink, pocket, color, white, red, stone, vegetable, green, large, shoe, scale, soft, steak, beer, knife, marmoset, carpet, bowl, australia, corn, fridge, soap, coin, fruit, fork, cuba, france, italy, steel, wall, unite state, cup, hill, square, apple tree, shelf, friend house, space, this, place, apple, bag, river, blue, grass, cheese, mammal, bean, flat, utah, plastic, container, bug, live room, toilet, cabinet, table, furniture, lamp, pizza, dust, sand, hall, dictionary, rise, closet, door, floor, basement, dark, world, air, sheep, statue, metal, big, squirrel, alcohol, clock, round, top, wine, jar, put, duck, edible, wyom, land, rug, pot, bread, field, lawn, tin, oven, shade, egg, building, bee, resturant, wash, sock, bear, pantry, supermarket, roof, brown, cake, solid, dirt, handle, alaska, michigan, maryland, maine, delaware, kansa, department, pretty, decoration, stapler, apartment, bush, countryside, any large city, case, sunshine, corner, outdoors, stick, find house, find outside, winery, branch, polish, wax, generic, ground, eaten, neighbor house, usually

Community 2 (size is 302). write program, go concert, hear music, exercise, listen, go performance, take walk, entertain, run marathon, wait line, attend lecture, study, go walk, play basketball, fun, bore, wait table, go see film, go work, watch tv show, wake up morning, go fish, tell story, surf web, play football, go restaurant, visit museum, study subject, live life, go sport event, go play, play soccer, go jog, take shower, play ball, watch movie, watch film, stretch, play frisbee, go school, surprise, paint picture, go film, rest, listen radio, kiss, remember, housework, watch tv, attend school, play tennis, comfortable, take bus, conversation, talk, take course, learn, think, go run, sleep, hang out bar, plan vacation, go see play, attend class, go swim, ride bike, eat restaurant, stress, boredom, ticket, use television, entertainment, listen music, enjoyment, student, go movie, enlightenment, stand line, attend classical concert, death, play sport, effort, drive car, traveling, knowledge, teach, laugh joke, run, read book, education, take note, travel, go store, go sleep, tire, attention, fall asleep, run errand, patience, spend money, cry, pay bill, earn money, speak, fatigue, take break, drink alcohol, play chess, anger, read, curiosity, pay, drive, use computer, take film, smile, fiddle, wrestle, see new, dance, job, smart, play baseball, excite, attend rock concert, hear news, contemplate, pain, understand, stay healthy, research, learn new, sweat, headache, fart, read newspaper, sport, understand better, write story, transportation, fall down, practice, help, lose, close eye, satisfaction, time, answer question, perform, everyone, go somewhere, play card, go, sex, wait, buy ticket, gain knowledge, interest, sit down, surf, teacher, happiness, exhaustion, sit chair, laugh, relax, waste time, pleasure, relaxation, procreate, funny, win, go mall, flirt, pass time, go home, love else, drunk, sing, buy beer, internet, like, date, play game, activity, jog, quiet, skill, hobby, tiredness, communication, drink coffee, read magazine, good time, good health, play hockey, eat ice cream, learn language, go zoo, go internet, cash, read child, enjoy yourself, see movie, emotion, clean house, fit, view video, play poker, excitement, fly airplane, ride horse, stay bed, happy,

find information, fear, go vacation, breathe, recreation, enjoy, ride bicycle, health, communicate, make money, become tire, lose weight, jump up down, watch television, healthy, know, learn subject, joy, stand up, information, read letter, jump rope, celebrate, sadness, watch musician perform, feel better, compete, feel good, accident, stay fit, injury, ride, play piano, learn world, see exhibit, release energy, see art, see excite story, orgasm, laughter, express yourself, discover truth, see favorite show, go party, competition, express information, climb mountain, attend meet, fly kite, examine, meet friend, read news, shock, return work, see band, visit art gallery, earn live, watch television show, socialize, create art, crossword puzzle, enjoy film, go pub, feel happy, play lacrosse, socialis, physical activity, get, make person laugh, make friend, chat friend, meet person, meet interest person, get drunk, friend over, get exercise, get tire, enjoy company friend, play game friend, get physical activity, go opus, get shape, sit quietly, do it, get fit, teach other person, entertain person, see person play game

Community 3 (size is 275). something, man, person, type, train, work, word, pant, love, library, bath, school, arm, human, class, walk, drink, it, dream, shower, child, smoke, gym, movie, sit, ball, eat food, mother, party, clean, lunch, street, trouble, play, bus, plan, eat, bridge, nothing, computer, line, buy, tv, car, vehicle, dog, music, dress, live, one, turn, fish, baby, hurt, game, hospital, hide, girl, muscle, woman, cold, family, shop, letter, eat dinner, concert, call, electricity, eye, see, story, nose, smell, stand, die, money, bill, leg, everything, television, take bath, band, drink water, paint, hike, lie, friend, hot, kill, swim, break, foot, verb, camp, den, we, fight, telephone, audience, drop, rain, body, use, write, doll, wheel, name, nice, book, museum, pool, instrument, bad, show, wind, hand, pee, stop, road, seat, boat, train station, war, wet, cell, bicycle, need, life, burn, sound, good, fire, news, finger, feel, dangerous, ski, expensive, leave, number, heavy, map, piano, club, theatre, god, care, airplane, watch, phone, radio, tool, mouth, doctor, theater, lot, hair, make, noise, measure, shape, climb, wash hand, bar, view, tooth, peace, kid, boy, record, find, song, meet, not, sofa, cut, page, company, bite, science, college, open, warm, high, birthday, university, roll, mean, general, act, heat, cool, dive, skin, art, noun, hard, important, toy, ring, crowd, draw, thing, energy, kill person, little, change, ear, alive, move, wave, look, voice, face, event, long, carry, fly, test, hear, organization, jump, business, action, pass, fall, bell, head, sign, count, end, group, bullet, degree, note, card, machine, lay, gas, circle, point, useful, adjective, be, steam, bike, side, motion, classroom, out, transport, step, part, course, power, same, stage, comfort, trip, edge, grow, board, race, flea, punch, cool off, skate, movement, away, short, many person, singular, general term, unit, generic term

9.2.4 Betweenness

The algorithm used is the one implemented in `igraph` which is based on [5]. The idea of the algorithm is described below; it is taken from `igraph` documentation online.

> The idea is that the betweenness of the edges connecting two communities is typically high, as many of the shortest paths between nodes in separate communities go through them. So we gradually remove the edge with highest betweenness from the network, and recalculate edge betweenness after every removal. This way sooner or later the network falls off to two components, then after a while one of these components falls off to two smaller components, etc. until all edges are removed. This is a divisive hierarchical approach, the result is a dendrogram.

The algorithm has complexity $\mathcal{O}\left(|V||E|^2\right)$, as the betweenness calculation requires $\mathcal{O}\left(|V||E|\right)$ time and we do it $|E|-1$ times. Hence, we applied the algorithm only on the subgraph induced by the vertices with maximum coreness [2] .

One execution of the algorithm in the subgraph took about $8,671.19$ seconds of computation time. The algorithm found 42 communities and the modularity achieved was 0.268508.

Edge Betweenness: Communities in the Inner-Most Core

We have the following communities.

[2] Recall that the subgraph has no self-loops.

Community 1 (size is 423). something, person, train, town, rock, beach, tree, soup, weasel, word, library, school, kitten, arm, human, plane, class, it, paper, bed, dirty, chicken, gym, office build, ball, box, object, mother, coffee, candle, street, fungus, park, snake, wood, bus, bridge, cloud, computer, line, milk, drawer, storage, car, vehicle, dog, zoo, bottle, live, one, material, chair, cat, hat, country, market, house, lake, hotel, plant, hospital, bank, girl, woman, animal, church, cold, family, moon, pet, cook, shop, letter, bird, concert, bathroom, city, water, yard, desk, office, home, bat, couch, kitchen, lizard, build, restaurant, spoon, butter, beautiful, key, eye, nose, stand, well, pen, bill, snow, weather, everything, mouse, magazine, hole, nature, band, bald eagle, nest, crab, paint, ficus, sea, ocean, sun, sky, food, grape, bedroom, horse, store, hot, airport, sugar, grocery store, basket, hold, kill, foot, refrigerator, newspaper, rice, surface, liquid, meadow, camp, window, oil, cover, plate, dinner, den, cow, earth, garage, garden, poop, potato, outside, frog, napkin, light, salad, fox, forest, glass, cupboard, telephone, marmot, mountain, salt, motel, drop, bone, meat, bookstore, rain, body, use, ferret, small dog, cloth, factory, bottle wine, doll, pencil, wheel, name, nice, book, black, instrument, show, trash, can, a, wind, hand, pee, picture, road, seat, boat, clothe, dish, train station, war, mall, wet, flower, wallet, room, cell, small, bicycle, new york, farm, sink, pocket, color, white, red, stone, vegetable, green, burn, large, shoe, scale, soft, steak, fire, beer, finger, knife, dangerous, carpet, bowl, corn, fridge, soap, expensive, coin, number, fruit, heavy, map, fork, steel, piano, wall, theatre, cup, hill, square, shelf, god, friend house, airplane, space, phone, this, place, radio, tool, apple, mouth, bag, doctor, theater, river, blue, grass, cheese, mammal, bean, lot, hair, make, measure, flat, utah, plastic, container, bar, bug, view, live room, toilet, tooth, cabinet, table, furniture, lamp, pizza, dust, sand, hall, rise, closet, boy, door, floor, not, basement, sofa, cut, page, company, bite, dark, college, world, air, sheep, statue, metal, open, warm, big, high, squirrel, alcohol, university, roll, general, clock, round, heat, cool, skin, art, noun, top, wine, jar, hard, put, duck, toy, ring, crowd, draw, edible, thing, land, rug, pot, little, change, ear, alive, bread, field, wave, face, tin, oven, long, shade, carry, fly, egg, building, bee, business, pass, resturant, wash, sock, bear, bell, head, sign, pantry, bullet, degree, note, card, supermarket, machine, gas, roof, brown, circle, cake, solid, dirt, point, useful, handle, adjective, maine, department, be, steam, pretty, side, decoration, stapler, classroom, transport, step, apartment, part, course, countryside, power, same, stage, any large city, edge, case, grow, board, flea, corner, short, stick, singular, find house, find outside, winery, branch, polish, wax, general term, generic, ground, eaten, neighbor house, usually, unit, generic term

Community 2 (size is 406). man, type, work, write program, go concert, hear music, exercise, love, bath, listen, go performance, take walk, walk, entertain, run marathon, wait line, attend lecture, drink, study, go walk, play basketball, fun, bore, wait table, go see film, go work, watch tv show, wake up morning, dream, shower, child, smoke, go fish, tell story, surf web, play football, movie, go restaurant, visit museum, study subject, live life, go sport event, go play, sit, play soccer, go jog, take shower, play ball, watch movie, watch film, stretch, play frisbee, go school, surprise, paint picture, go film, party, rest, listen radio, kiss, remember, housework, clean, lunch, watch tv, attend school, play tennis, trouble, comfortable, play, take bus, conversation, talk, take course, learn, plan, think, go run, sleep, hang out bar, plan vacation, go see play, eat, attend class, go swim, ride bike, nothing, buy, eat restaurant, tv, stress, boredom, ticket, music, use television, dress, turn, entertainment, listen music, enjoyment, fish, baby, hurt, game, student, muscle, go movie, enlightenment, stand line, attend classical concert, death, play sport, effort, drive car, traveling, knowledge, teach, call, laugh joke, run, read book, education, take note, travel, electricity, go store, see, story, smell, go sleep, tire, attention, die, fall asleep, money, leg, run errand, patience, spend money, cry, pay bill, earn money, television, speak, drink water, fatigue, take break, hike, drink alcohol, lie, play chess, friend, anger, read, curiosity, pay, swim, break, verb, drive, use computer, take film, smile, fiddle, we, wrestle, see new, dance, fight, job, smart, play baseball, excite, attend rock concert, hear news, contemplate, pain, audience, understand, write, stay healthy, research, learn new, sweat, headache, fart, read newspaper, sport, understand better, bad, write story, stop, transportation, fall down, practice, help, lose, close eye, satisfaction, time, answer question, perform, need, everyone, go somewhere, life, sound, good, play card, go, sex, wait, buy ticket, gain knowledge, news, interest, feel, sit down, ski, surf, teacher, leave, happiness, exhaustion, sit chair, laugh, relax, waste time, pleasure, relaxation, care, procreate, watch, funny, win, go mall, flirt, pass time, noise, shape, climb, go home, love else, drunk, peace, sing, buy beer, internet, kid, like, date, record, find, song, play game, meet, activity, science, jog, quiet, skill, hobby, birthday, tiredness,

mean, communication, drink coffee, read magazine, good time, good health, act, play hockey, eat ice cream, learn language, dive, go zoo, go internet, cash, important, read child, enjoy yourself, see movie, energy, kill person, emotion, clean house, fit, view video, play poker, excitement, move, fly airplane, ride horse, stay bed, look, voice, event, happy, find information, fear, go vacation, breathe, recreation, test, enjoy, hear, jump, ride bicycle, health, communicate, make money, become tire, action, fall, lose weight, jump up down, watch television, count, healthy, end, know, learn subject, joy, stand up, information, read letter, lay, jump rope, celebrate, sadness, bike, watch musician perform, motion, feel better, compete, out, feel good, accident, stay fit, injury, ride, play piano, learn world, see exhibit, release energy, see art, see excite story, comfort, orgasm, trip, laughter, express yourself, discover truth, see favorite show, go party, competition, express information, climb mountain, attend meet, fly kite, examine, race, meet friend, read news, shock, return work, see band, visit art gallery, earn live, punch, cool off, watch television show, socialize, skate, movement, create art, crossword puzzle, enjoy film, go pub, feel happy, play lacrosse, socialis, away, physical activity, get, many person, make person laugh, make friend, chat friend, meet person, meet interest person, get drunk, friend over, get exercise, get tire, enjoy company friend, play game friend, get physical activity, go opus, get shape, sit quietly, do it, get fit, teach other person, entertain person, see person play game

Community 3 (size is 1). monkey

Community 4 (size is 1). pant

Community 5 (size is 1). beaver

Community 6 (size is 1). state

Community 7 (size is 1). wiener dog

Community 8 (size is 1). eat food

Community 9 (size is 1). hide

Community 10 (size is 1). eat dinner

Community 11 (size is 1). shark

Community 12 (size is 1). rosebush

Community 13 (size is 1). sloth

Community 14 (size is 1). take bath

Community 15 (size is 1). anemone

Community 16 (size is 1). lemur

Community 17 (size is 1). museum

Community 18 (size is 1). pool

Community 19 (size is 1). canada

Community 20 (size is 1). wild

Community 21 (size is 1). marmoset

Community 22 (size is 1). australia

Community 23 (size is 1). cuba

Community 24 (size is 1). france

Community 25 (size is 1). italy

Community 26 (size is 1). club

Community 27 (size is 1). unite state

Community 28 (size is 1). apple tree

Community 29 (size is 1). wash hand

Community 30 (size is 1). dictionary

Community 31 (size is 1). wyom

Community 32 (size is 1). lawn

Community 33 (size is 1). organization

Community 34 (size is 1). group

Community 35 (size is 1). alaska

Community 36 (size is 1). michigan

Community 37 (size is 1). maryland

Community 38 (size is 1). delaware

Community 39 (size is 1). kansa

Community 40 (size is 1). bush

Community 41 (size is 1). sunshine

Community 42 (size is 1). outdoors

9.2.5 Fast Greedy

The algorithm used is the one implemented in igraph which is based on [2]. According to igraph version 0.6.1 which was used at the time of the writing, some improvements mentioned in [14] have also been implemented. Table 9.14 presents the results when we apply the algorithm for subgraphs in which we include vertices with successively lower coreness and allow positive polarity on the edges only.

Fast Greedy: Communities in the Inner-Most Core

We have the following communities.

Community 1 (size is 20). class, gym, turn, woman, smell, take bath, pool, pee, wet, radio, shape, view, science, art, change, wave, test, pass, wash, course

Community 2 (size is 440). something, town, rock, beach, tree, monkey, soup, weasel, pant, library, school, kitten, arm, human, plane, beaver, it, paper, bed, dirty, chicken, state, office build, wiener dog, ball, box, object, mother, coffee, candle, street, fungus, park, snake, wood, bus, eat, bridge, cloud, computer, line, milk, drawer, storage, car, vehicle, dog, zoo, bottle, live, one, material, chair, cat, hat, country, market, house, fish, lake, baby, hotel, plant, game, hospital, bank, hide, girl, animal, church, cold, family, moon, pet, cook, shop, letter, bird, bathroom, city, shark, water, rosebush, yard, desk, office, home, sloth, bat, couch, kitchen, lizard, build, restaurant, spoon, butter, beautiful, key, electricity, eye, nose, stand, well, pen, bill, snow, weather, leg, everything, mouse, magazine, hole, nature, bald eagle, nest, crab, paint, ficus, sea, anemone, ocean, sun, sky, food, grape, bedroom, horse, store, hot, airport, sugar, grocery store, basket, hold, foot, refrigerator, newspaper, rice, surface, liquid, meadow, camp, window, oil, cover, plate, dinner, den, cow, earth, garage, garden, poop, potato, outside, frog, napkin, light, salad, fox, forest, glass, cupboard, telephone, marmot, mountain, salt, motel, drop, bone, meat, bookstore, rain, body, use, ferret, small dog, cloth, factory, bottle wine, doll, pencil, wheel, lemur, name, nice, book, museum, black, canada, instrument, trash, can, a, wind, hand, picture, road, seat, boat, wild, clothe, dish, train station, war, mall, flower, wallet, room, cell, small, bicycle, new york, farm, sink, pocket, color, white, red, stone, vegetable, green, life, burn, large, shoe, scale, soft, steak, fire, beer, finger, knife, dangerous, marmoset, carpet, bowl, australia, corn, fridge, soap, expensive, coin, number, fruit, heavy, map, fork, cuba, france, italy, steel, piano, wall, theatre, unite state, cup, hill, square, apple tree, shelf, god, friend house, airplane, space, phone, this, place, tool, apple, mouth, bag, doctor, theater, river, blue, grass, cheese, mammal, bean, lot, hair, make, measure, flat, utah, plastic, container, bar, bug, live room, toilet, tooth, cabinet, table, furniture, lamp, pizza, dust, sand, hall, dictionary, rise, closet, boy, door, floor, basement, sofa, cut, page, company, bite, dark, college, world, air, sheep, statue, metal, open, warm, big, high, squirrel, alcohol, university, roll, general, clock, round, heat, skin, noun, top, wine, jar, hard, put, duck, toy, ring, draw, edible, wyom, thing, land, rug, pot, little, ear, alive, bread, field, face, lawn, tin, oven, long, shade, carry, fly, egg, building, bee, business, resturant, sock, bear, bell, head, sign, pantry, bullet, degree, note, card, supermarket, machine, gas, roof, brown, circle, cake, solid, dirt, point, useful, handle, adjective, alaska, michigan, maryland, maine, delaware, kansa, department, be, steam, pretty, side, decoration, stapler, classroom, transport, step, apartment, part, bush, countryside, power, same, stage, any large city, comfort, edge, case, grow, board, sunshine, flea, corner, short, outdoors, stick, singular, find house, find outside, winery, branch, polish, wax, general term, generic, ground, eaten, neighbor house, usually, unit, generic term

Community 3 (size is 12). word, concert, band, kill, club, not, mean, cool, crowd, organization, end, group

Community 4 (size is 397). man, person, type, train, work, write program, go concert, hear music, exercise, love, bath, listen, go performance, take walk, walk, entertain, run marathon, wait line, attend lecture, drink, study, go walk, play basketball, fun, bore, wait table, go see film, go work, watch tv show, wake up morning, dream, shower, child, smoke, go fish, tell story, surf web, play football, movie, go restaurant, visit museum, study subject, live life, go sport event, go play, sit, play soccer, go jog, take shower, play ball, eat food, watch movie, watch film, stretch, play frisbee, go school, surprise, paint picture, go film, party, rest, listen radio, kiss, remember, housework, clean, lunch, watch tv, attend school, play tennis, trouble, comfortable, play, take bus, conversation, talk, take course, learn, plan, think, go run, sleep, hang out bar, plan vacation, go see play, attend class, go swim, ride bike, nothing, buy, eat restaurant, tv, stress, boredom, ticket, music, use television, dress, entertainment, listen music, enjoyment, hurt, student, muscle, go movie, enlightenment, stand line, attend classical concert, death, play sport, eat dinner, effort, drive car, traveling, knowledge, teach, call, laugh joke, run, read book, education, take note, travel, go store, see, story, go sleep, tire, attention, die, fall asleep, money, run errand, patience, spend money, cry, pay bill, earn money, television, speak, drink water, fatigue, take break, hike, drink alcohol, lie, play chess, friend, anger, read, curiosity, pay, swim, break, verb, drive, use computer, take film, smile, fiddle, we, wrestle, see new, dance, fight, job, smart, play baseball, excite, attend rock concert, hear news, contemplate, pain, audience, understand, write, stay healthy, research, learn new, sweat, headache, fart, read newspaper, sport, understand better, bad, show, write story, stop, transportation, fall down, practice, help, lose, close eye, satisfaction, time, answer question, perform, need, everyone, go somewhere, sound, good, play card, go, sex, wait, buy ticket, gain knowledge, news, interest, feel, sit down, ski, surf, teacher, leave, happiness, exhaustion, sit chair, laugh, relax, waste time, pleasure, relaxation, care, procreate, watch, funny, win, go mall, flirt, pass time, noise, climb, wash hand, go home, love else, drunk, peace, sing, buy beer, internet, kid, like, date, record, find, song, play game, meet, activity, jog, quiet, skill, hobby, birthday, tiredness, communication, drink coffee, read magazine, good time, good health, act, play hockey, eat ice cream, learn language, dive, go zoo, go internet, cash, important, read child, enjoy yourself, see movie, energy, kill person, emotion, clean house, fit, view video, play poker, excitement, move, fly airplane, ride horse, stay bed, look, voice, event, happy, find information, fear, go vacation, breathe, recreation, enjoy, hear, jump, ride bicycle, health, communicate, make money, become tire, action, fall, lose weight, jump up down, watch television, count, healthy, know, learn subject, joy, stand up, information, read letter, lay, jump rope, celebrate, sadness, bike, watch musician perform, motion, feel better, compete, out, feel good, accident, stay fit, injury, ride, play piano, learn world, see exhibit, release energy, see art, see excite story, orgasm, trip, laughter, express yourself, discover truth, see favorite show, go party, competition, express information, climb mountain, attend meet, fly kite, examine, race, meet friend, read news, shock, return work, see band, visit art gallery, earn live, punch, cool off, watch television show, socialize, skate, movement, create art, crossword puzzle, enjoy film, go pub, feel happy, play lacrosse, socialis, away, physical activity, get, many person, make person laugh, make friend, chat friend, meet person, meet interest person, get drunk, friend over, get exercise, get tire, enjoy company friend, play game friend, get physical activity, go opus, get shape, sit quietly, do it, get fit, teach other person, entertain person, see person play game

9.2.6 Multilevel

The algorithm used is the one implemented in `igraph` which is based on [1]. Table 9.15 presents the results when we apply the algorithm for subgraphs in which we include vertices with successively lower coreness and allow positive polarity on the edges only.

Multilevel: Communities in the Inner-Most Core

We have the following communities.

Community 1 (size is 92). soup, drink, chicken, eat food, coffee, lunch, eat, milk, bottle, market, fish, cook, shop, eat dinner, kitchen, restaurant, spoon, butter, eye, food, grape, sugar, grocery store, basket, hold, refrigerator, rice, liquid, oil, plate, dinner, potato, napkin, salad, glass, cupboard, salt, bone, meat, bottle wine, can, a, hand, dish, sink, white, vegetable, steak, beer, knife, bowl, corn,

fridge, soap, fruit, fork, cup, apple, mouth, cheese, bean, container, wash hand, tooth, cabinet, pizza, cut, open, alcohol, round, skin, wine, jar, edible, pot, ear, bread, face, tin, oven, egg, resturant, wash, head, pantry, supermarket, cake, steam, power, same, eaten, usually

Community 2 (size is 180). rock, beach, tree, monkey, weasel, pant, kitten, arm, beaver, smoke, state, wiener dog, fungus, park, snake, wood, bridge, cloud, nothing, dog, zoo, live, cat, hat, country, lake, baby, plant, hide, animal, cold, moon, pet, bird, shark, water, rosebush, yard, sloth, bat, lizard, beautiful, nose, well, snow, weather, everything, mouse, hole, nature, bald eagle, nest, crab, ficus, sea, anemone, ocean, sun, sky, horse, hot, meadow, camp, den, cow, earth, garden, poop, outside, frog, light, fox, forest, marmot, mountain, rain, body, ferret, small dog, lemur, nice, museum, black, canada, wind, wild, flower, small, new york, farm, color, red, stone, green, life, burn, large, soft, fire, dangerous, marmoset, australia, leave, heavy, cuba, france, italy, unite state, hill, apple tree, god, space, river, blue, grass, mammal, lot, hair, utah, bug, view, sand, dictionary, rise, bite, dark, science, world, air, sheep, statue, warm, big, high, squirrel, general, heat, cool, art, hard, duck, wyom, land, little, alive, field, lawn, long, shade, fly, bee, bear, gas, brown, solid, dirt, adjective, alaska, michigan, maryland, maine, delaware, kansa, be, pretty, bush, course, countryside, grow, sunshine, flea, short, outdoors, stick, find outside, branch, wax, generic, ground, generic term

Community 3 (size is 184). something, type, town, word, library, school, human, plane, class, it, paper, bed, dirty, office build, sit, box, object, mother, candle, street, bus, computer, line, drawer, storage, car, vehicle, turn, material, chair, house, hotel, game, hospital, bank, girl, church, family, letter, bathroom, city, desk, office, home, couch, build, key, electricity, stand, pen, bill, magazine, band, paint, bedroom, store, airport, newspaper, surface, window, cover, garage, telephone, motel, bookstore, use, write, cloth, factory, doll, pencil, name, book, instrument, trash, picture, road, seat, boat, clothe, train station, mall, wallet, room, cell, pocket, shoe, scale, finger, carpet, expensive, coin, number, map, steel, piano, wall, club, square, shelf, friend house, airplane, phone, this, place, radio, tool, bag, doctor, theater, make, measure, flat, plastic, bar, live room, toilet, table, furniture, lamp, dust, hall, closet, boy, door, floor, basement, sofa, page, company, college, metal, university, clock, noun, top, put, toy, ring, crowd, draw, thing, rug, change, carry, test, organization, building, business, sock, bell, sign, group, bullet, degree, note, card, machine, roof, circle, point, useful, handle, department, side, decoration, stapler, classroom, apartment, part, stage, any large city, comfort, edge, case, board, corner, singular, find house, winery, polish, general term, neighbor house, unit

Community 4 (size is 149). man, train, work, exercise, take walk, walk, run marathon, go walk, play basketball, wait table, go work, wake up morning, shower, child, gym, play football, play soccer, go jog, take shower, play ball, ball, stretch, play frisbee, housework, clean, play tennis, plan, go run, go swim, ride bike, stress, dress, one, hurt, muscle, woman, death, play sport, effort, drive car, traveling, run, travel, go store, smell, tire, die, leg, run errand, earn money, drink water, fatigue, hike, kill, swim, break, foot, verb, drive, wrestle, fight, play baseball, pain, drop, stay healthy, wheel, sweat, pool, sport, bad, pee, stop, transportation, fall down, practice, lose, war, wet, bicycle, go somewhere, go, ski, exhaustion, win, shape, climb, not, activity, jog, roll, tiredness, mean, drink coffee, good health, play hockey, dive, energy, kill person, clean house, fit, move, ride horse, wave, fear, jump, ride bicycle, health, become tire, action, pass, fall, lose weight, jump up down, count, healthy, end, stand up, jump rope, bike, motion, feel better, compete, out, accident, transport, stay fit, injury, ride, step, release energy, trip, competition, climb mountain, race, shock, return work, earn live, punch, cool off, skate, movement, play lacrosse, away, physical activity, get exercise, get tire, get physical activity, get shape, get fit

Community 5 (size is 264). person, write program, go concert, hear music, love, bath, listen, go performance, entertain, wait line, attend lecture, study, fun, bore, go see film, watch tv show, dream, go fish, tell story, surf web, movie, go restaurant, visit museum, study subject, live life, go sport event, go play, watch movie, watch film, go school, surprise, paint picture, go film, party, rest, listen radio, kiss, remember, watch tv, attend school, trouble, comfortable, play, take bus, conversation, talk, take course, learn, think, sleep, hang out bar, plan vacation, go see play, attend class, buy, eat restaurant, tv, boredom, ticket, music, use television, entertainment, listen music,

enjoyment, student, go movie, enlightenment, stand line, attend classical concert, concert, knowledge, teach, call, laugh joke, read book, education, take note, see, story, go sleep, attention, fall asleep, money, patience, spend money, cry, pay bill, television, speak, take bath, take break, drink alcohol, lie, play chess, friend, anger, read, curiosity, pay, use computer, take film, smile, fiddle, we, see new, dance, job, smart, excite, attend rock concert, hear news, contemplate, audience, understand, research, learn new, headache, fart, read newspaper, understand better, show, write story, help, close eye, satisfaction, time, answer question, perform, need, everyone, sound, good, play card, sex, wait, buy ticket, gain knowledge, news, interest, feel, sit down, surf, teacher, happiness, sit chair, laugh, theatre, relax, waste time, pleasure, relaxation, care, procreate, watch, funny, go mall, flirt, pass time, noise, go home, love else, drunk, peace, sing, buy beer, internet, kid, like, date, record, find, song, play game, meet, quiet, skill, hobby, birthday, communication, read magazine, good time, act, eat ice cream, learn language, go zoo, go internet, cash, important, read child, enjoy yourself, see movie, emotion, view video, play poker, excitement, fly airplane, stay bed, look, voice, event, happy, find information, go vacation, breathe, recreation, enjoy, hear, communicate, make money, watch television, know, learn subject, joy, information, read letter, lay, celebrate, sadness, watch musician perform, feel good, play piano, learn world, see exhibit, see art, see excite story, orgasm, laughter, express yourself, discover truth, see favorite show, go party, express information, attend meet, fly kite, examine, meet friend, read news, see band, visit art gallery, watch television show, socialize, create art, crossword puzzle, enjoy film, go pub, feel happy, socialis, get, many person, make person laugh, make friend, chat friend, meet person, meet interest person, get drunk, friend over, enjoy company friend, play game friend, go opus, sit quietly, do it, teach other person, entertain person, see person play game

9.2.7 Label Propagation

The algorithm used is the one implemented in `igraph` which is based on [9]. Table 9.16 presents the results when we apply the algorithm for subgraphs in which we include vertices with successively lower coreness and allow positive polarity on the edges only.

Label Propagation: Communities in the Inner-Most Core

The entire core is one big community. This is the result in all of our runs.

9.2.8 InfoMAP

The algorithm used is the one implemented in `igraph` which is based on [12]; see also [11]. Table 9.17 presents the results when we apply the algorithm for subgraphs in which we include vertices with successively lower coreness and allow edges with positive polarity only. The algorithm can exhibit in some cases wild variations both in terms of the computed communities as well as of the induced modularity.

InfoMAP: Communities in the Inner-Most Core

The entire core is one big community. This is the result in all of our runs.

Table 9.10: Overall comparison of the community finding algorithms implemented in `igraph`. We use π to indicate coreness. We can see the average number $\overline{\kappa}$ of communities found per run, as well as the average modularity $\overline{\mu}$ achieved by each algorithm. Bold entries in the columns with the modularities indicate the maximum value achieved among all algorithms per row; that is per subgraph induced by vertices that have coreness at least a certain lower bound value. The best values for the modularity are achieved by the spinglass algorithm and the multilevel algorithm. We do not see a result for spinglass in the last row (that is coreness $\geqslant 2$) because the implementation of the algorithm expects a connected graph. Hence we need to run the algorithm in each connected component. However, the difference in terms of actual computation time is huge compared to the multilevel algorithm.

subgraph properties				spin glass		eigen-vectors		walktrap		between-ness		fast greedy		multi-level		label propagation		infoMAP	
$\pi \geqslant$	$\|V\|$	$\|E\|$	$\|C\|$	$\overline{\kappa}$	$\overline{\mu}$	$\overline{\kappa}$	$\overline{\mu}$	$\overline{\kappa}$	$\overline{\mu}$	$\overline{\kappa}$	$\overline{\mu}$	$\overline{\kappa}$	$\overline{\mu}$	$\overline{\kappa}$	$\overline{\mu}$	$\overline{\kappa}$	$\overline{\mu}$	$\overline{\kappa}$	$\overline{\mu}$
26	869	20526	1	6	0.323	4	0.304	3	0.275	42	0.268508	4	0.287	5	0.322	1.00	0.000	1.0	0.000
25	1167	27810	1	7	0.309	5	0.300	3	0.282	–	–	3	0.295	6	0.320	1.01	0.003	1.0	0.000
24	1358	32314	1	8	0.315	4	0.290	3	0.282	–	–	4	0.285	7	0.321	1.06	0.017	1.0	0.000
23	1514	35870	1	8	0.316	3	0.283	4	0.275	–	–	4	0.284	6	0.322	1.00	0.000	1.0	0.000
22	1709	40099	1	6	0.320	4	0.287	3	0.287	–	–	5	0.283	7	0.323	1.01	0.003	1.0	0.000
21	1865	43330	1	10	0.320	4	0.286	3	0.283	–	–	4	0.292	8	0.314	1.00	0.000	1.0	0.000
20	2007	46145	1	7	0.328	4	0.290	4	0.285	–	–	3	0.291	7	0.321	1.03	0.008	1.0	0.000
19	2173	49265	1	12	0.323	5	0.293	3	0.274	–	–	4	0.294	7	0.321	1.01	0.003	1.0	0.000
18	2384	53011	1	10	0.326	4	0.292	4	0.292	–	–	6	0.287	8	0.329	1.01	0.003	1.0	0.000
17	2617	56939	1	12	0.328	5	0.293	5	0.279	–	–	6	0.285	9	0.331	1.00	0.000	2.0	0.009
16	2847	60583	1	10	0.331	4	0.279	4	0.285	–	–	6	0.295	6	0.331	1.02	0.006	2.0	0.012
15	3105	64412	1	11	0.334	4	0.280	4	0.284	–	–	6	0.292	7	0.336	1.00	0.000	2.0	0.012
14	3407	68613	1	14	0.337	5	0.272	6	0.306	–	–	5	0.303	9	0.348	1.01	0.003	3.2	0.039
13	3746	72978	1	12	0.341	5	0.275	5	0.291	–	–	6	0.300	9	0.344	1.00	0.000	2.0	0.014
12	4160	77882	1	14	0.344	6	0.283	8	0.290	–	–	10	0.292	10	0.349	1.00	0.000	3.3	0.042
11	4634	83039	1	7	0.351	5	0.303	8	0.320	–	–	12	0.298	10	0.348	1.01	0.003	18.0	0.188
10	5182	88462	1	9	0.357	4	0.281	10	0.311	–	–	10	0.307	10	0.362	1.00	0.000	34.5	0.269
9	5883	94709	1	11	0.363	4	0.275	13	0.319	–	–	14	0.319	9	0.364	1.02	0.003	55.9	0.347
8	6750	101564	1	10	0.372	4	0.273	17	0.323	–	–	13	0.319	10	0.368	1.00	0.000	81.1	0.345
7	7904	109561	1	7	0.384	2	0.267	15	0.328	–	–	18	0.326	10	0.375	1.02	0.002	117.1	0.348
6	9392	118389	1	11	0.393	4	0.283	17	0.341	–	–	17	0.334	10	0.387	1.00	0.000	166.8	0.350
5	11483	128731	1	8	0.389	3	0.290	31	0.339	–	–	24	0.333	12	0.392	1.00	0.000	251.6	0.353
4	14864	142112	1	12	0.413	3	0.290	54	0.334	–	–	37	0.339	11	0.402	1.12	0.000	407.9	0.350
3	21812	162691	1	15	0.423	4	0.288	223	0.343	–	–	61	0.373	13	0.419	2.63	0.006	747.5	0.348
2	41659	201678	4	–	–	12	0.248	–	–	–	–	252	0.409	25	0.449	12.71	0.010	1753.5	0.351

	spin glass	eigen-vectors	walktrap	between-ness	fast greedy	multi-level	label propagation	infoMAP
duration/run (secs)	22,443.0	47.5	572.4	8,671.2	625.6	6.9	20.7	1,337.6

Table 9.11: Applying the Spin Glass algorithm for community finding implemented in `igraph` by successively including vertices with lower coreness on the undirected graph induced by the assertions with positive polarity (self-loops are removed). In every row we have the number of vertices and the number of edges of each such subgraph together with the number of components ($|C|$) that we find in that subgraph. The next three columns present the number of communities found by the algorithm; the average among all runs, the minimum, and the maximum. The next three columns present the modularity achieved by the algorithm due to the cut induced by the communities; the average among all runs, the minimum, and the maximum. The entire computation lasted $22,442.95$ seconds for a single run.

coreness	$\|V\|$	$\|E\|$	$\|C\|$	communities found			modularity		
				avg	min	max	avg	min	max
$\geqslant 26$	869	20526	1	6.000	6	6	0.323492	0.323492	0.323492
$\geqslant 25$	1167	27810	1	7.000	7	7	0.309399	0.309399	0.309399
$\geqslant 24$	1358	32314	1	8.000	8	8	0.315379	0.315379	0.315379
$\geqslant 23$	1514	35870	1	8.000	8	8	0.316294	0.316294	0.316294
$\geqslant 22$	1709	40099	1	6.000	6	6	0.320150	0.320150	0.320150
$\geqslant 21$	1865	43330	1	10.000	10	10	0.320400	0.320400	0.320400
$\geqslant 20$	2007	46145	1	7.000	7	7	0.328107	0.328107	0.328107
$\geqslant 19$	2173	49265	1	12.000	12	12	0.323378	0.323378	0.323378
$\geqslant 18$	2384	53011	1	10.000	10	10	0.326275	0.326275	0.326275
$\geqslant 17$	2617	56939	1	12.000	12	12	0.327978	0.327978	0.327978
$\geqslant 16$	2847	60583	1	10.000	10	10	0.331368	0.331368	0.331368
$\geqslant 15$	3105	64412	1	11.000	11	11	0.334370	0.334370	0.334370
$\geqslant 14$	3407	68613	1	14.000	14	14	0.337464	0.337464	0.337464
$\geqslant 13$	3746	72978	1	12.000	12	12	0.340777	0.340777	0.340777
$\geqslant 12$	4160	77882	1	14.000	14	14	0.343900	0.343900	0.343900
$\geqslant 11$	4634	83039	1	7.000	7	7	0.351369	0.351369	0.351369
$\geqslant 10$	5182	88462	1	9.000	9	9	0.357447	0.357447	0.357447
$\geqslant 9$	5883	94709	1	11.000	11	11	0.362553	0.362553	0.362553
$\geqslant 8$	6750	101564	1	10.000	10	10	0.371669	0.371669	0.371669
$\geqslant 7$	7904	109561	1	7.000	7	7	0.383777	0.383777	0.383777
$\geqslant 6$	9392	118389	1	11.000	11	11	0.393008	0.393008	0.393008
$\geqslant 5$	11483	128731	1	8.000	8	8	0.388912	0.388912	0.388912
$\geqslant 4$	14864	142112	1	12.000	12	12	0.413350	0.413350	0.413350
$\geqslant 3$	21812	162691	1	15.000	15	15	0.423025	0.423025	0.423025
$\geqslant 2$	41659	201678	4	–	–	–	–	–	–

Table 9.12: Applying the leading eigenvector algorithm for community finding implemented in `igraph` by successively including vertices with lower coreness on the undirected graph induced by the assertions with positive polarity (self-loops are removed). In every row we have the number of vertices and the number of edges of each such subgraph together with the number of components ($|C|$) that we find in that subgraph. The next three columns present the number of communities found by the algorithm; the average among all runs, the minimum, and the maximum. The next three columns present the modularity achieved by the algorithm due to the cut induced by the communities; the average among all runs, the minimum, and the maximum. The entire computation lasted 94.98 seconds for 2 runs; that is about 47.49 seconds per run.

coreness	$\|V\|$	$\|E\|$	$\|C\|$	communities found			modularity		
				avg	min	max	avg	min	max
$\geqslant 26$	869	20526	1	4.000	4	4	0.304160	0.304160	0.304160
$\geqslant 25$	1167	27810	1	5.000	5	5	0.300161	0.300161	0.300161
$\geqslant 24$	1358	32314	1	4.000	4	4	0.289870	0.289870	0.289870
$\geqslant 23$	1514	35870	1	3.000	3	3	0.282888	0.282888	0.282888
$\geqslant 22$	1709	40099	1	4.000	4	4	0.287385	0.287385	0.287385
$\geqslant 21$	1865	43330	1	4.000	4	4	0.286488	0.286488	0.286488
$\geqslant 20$	2007	46145	1	4.000	4	4	0.289613	0.289613	0.289613
$\geqslant 19$	2173	49265	1	5.000	5	5	0.293102	0.293102	0.293102
$\geqslant 18$	2384	53011	1	4.000	4	4	0.292260	0.292260	0.292260
$\geqslant 17$	2617	56939	1	5.000	5	5	0.293102	0.293102	0.293102
$\geqslant 16$	2847	60583	1	4.000	4	4	0.278867	0.278867	0.278867
$\geqslant 15$	3105	64412	1	4.000	4	4	0.280229	0.280229	0.280229
$\geqslant 14$	3407	68613	1	5.000	5	5	0.271588	0.271588	0.271588
$\geqslant 13$	3746	72978	1	5.000	5	5	0.274841	0.274841	0.274841
$\geqslant 12$	4160	77882	1	6.000	6	6	0.282679	0.282679	0.282679
$\geqslant 11$	4634	83039	1	5.000	5	5	0.302635	0.302635	0.302635
$\geqslant 10$	5182	88462	1	4.000	4	4	0.281035	0.281035	0.281035
$\geqslant 9$	5883	94709	1	4.000	4	4	0.275202	0.275202	0.275202
$\geqslant 8$	6750	101564	1	4.000	4	4	0.273375	0.273375	0.273375
$\geqslant 7$	7904	109561	1	2.000	2	2	0.266965	0.266965	0.266965
$\geqslant 6$	9392	118389	1	4.000	4	4	0.282719	0.282719	0.282719
$\geqslant 5$	11483	128731	1	3.000	3	3	0.289544	0.289544	0.289544
$\geqslant 4$	14864	142112	1	3.000	3	3	0.290085	0.290085	0.290085
$\geqslant 3$	21812	162691	1	4.000	4	4	0.287835	0.287835	0.287835
$\geqslant 2$	41659	201678	4	12.000	12	12	0.247697	0.247697	0.247697

Table 9.13: Applying the Walktrap algorithm for community finding implemented in `igraph` by successively including vertices with lower coreness on the undirected graph induced by the assertions with positive polarity (self-loops are removed). We use 5 steps for every random walk generated throughout all our runs. In every row we have the number of vertices and the number of edges of each such subgraph together with the number of components ($|C|$) that we find in that subgraph. The next three columns present the number of communities found by the algorithm; the average among all runs, the minimum, and the maximum. The next three columns present the modularity achieved by the algorithm due to the cut induced by the communities; the average among all runs, the minimum, and the maximum. The entire computation lasted $1,144.71$ seconds for 2 runs.

coreness	$\|V\|$	$\|E\|$	$\|C\|$	communities found			modularity		
				avg	min	max	avg	min	max
$\geqslant 26$	869	20526	1	3.000	3	3	0.274815	0.274815	0.274815
$\geqslant 25$	1167	27810	1	3.000	3	3	0.282369	0.282369	0.282369
$\geqslant 24$	1358	32314	1	3.000	3	3	0.281649	0.281649	0.281649
$\geqslant 23$	1514	35870	1	4.000	4	4	0.275340	0.275340	0.275340
$\geqslant 22$	1709	40099	1	3.000	3	3	0.286858	0.286858	0.286858
$\geqslant 21$	1865	43330	1	3.000	3	3	0.283172	0.283172	0.283172
$\geqslant 20$	2007	46145	1	4.000	4	4	0.284822	0.284822	0.284822
$\geqslant 19$	2173	49265	1	3.000	3	3	0.273657	0.273657	0.273657
$\geqslant 18$	2384	53011	1	4.000	4	4	0.291989	0.291989	0.291989
$\geqslant 17$	2617	56939	1	5.000	5	5	0.279475	0.279475	0.279475
$\geqslant 16$	2847	60583	1	4.000	4	4	0.285183	0.285183	0.285183
$\geqslant 15$	3105	64412	1	4.000	4	4	0.284449	0.284449	0.284449
$\geqslant 14$	3407	68613	1	6.000	6	6	0.306199	0.306199	0.306199
$\geqslant 13$	3746	72978	1	5.000	5	5	0.291219	0.291219	0.291219
$\geqslant 12$	4160	77882	1	8.000	8	8	0.290368	0.290368	0.290368
$\geqslant 11$	4634	83039	1	8.000	8	8	0.320025	0.320025	0.320025
$\geqslant 10$	5182	88462	1	10.000	10	10	0.311006	0.311006	0.311006
$\geqslant 9$	5883	94709	1	13.000	13	13	0.318720	0.318720	0.318720
$\geqslant 8$	6750	101564	1	17.000	17	17	0.322721	0.322721	0.322721
$\geqslant 7$	7904	109561	1	15.000	15	15	0.327759	0.327759	0.327759
$\geqslant 6$	9392	118389	1	17.000	17	17	0.340760	0.340760	0.340760
$\geqslant 5$	11483	128731	1	31.000	31	31	0.338872	0.338872	0.338872
$\geqslant 4$	14864	142112	1	54.000	54	54	0.333880	0.333880	0.333880
$\geqslant 3$	21812	162691	1	223.000	223	223	0.342930	0.342930	0.342930
$\geqslant 2$	42576	208346	4	–	–	–	–	–	–

Table 9.14: Applying the Fast Greedy algorithm for community finding implemented in `igraph` by successively including vertices with lower coreness on the undirected graph induced by the assertions with positive polarity (self-loops are removed). In every row we have the number of vertices and the number of edges of each such subgraph together with the number of components ($|C|$) that we find in that subgraph. The next three columns present the number of communities found by the algorithm; the average among all runs, the minimum, and the maximum. The next three columns present the modularity achieved by the algorithm due to the cut induced by the communities; the average among all runs, the minimum, and the maximum. The entire computation lasted about $1,251.1$ seconds for 2 runs; that is about 625.55 seconds per run.

coreness	$\|V\|$	$\|E\|$	$\|C\|$	communities found			modularity		
				avg	min	max	avg	min	max
$\geqslant 26$	869	20526	1	4.000	4	4	0.286729	0.286729	0.286729
$\geqslant 25$	1167	27810	1	3.000	3	3	0.294925	0.294925	0.294925
$\geqslant 24$	1358	32314	1	4.000	4	4	0.285080	0.285080	0.285080
$\geqslant 23$	1514	35870	1	4.000	4	4	0.283817	0.283817	0.283817
$\geqslant 22$	1709	40099	1	5.000	5	5	0.283268	0.283268	0.283268
$\geqslant 21$	1865	43330	1	4.000	4	4	0.292439	0.292439	0.292439
$\geqslant 20$	2007	46145	1	3.000	3	3	0.291441	0.291441	0.291441
$\geqslant 19$	2173	49265	1	4.000	4	4	0.294087	0.294087	0.294087
$\geqslant 18$	2384	53011	1	6.000	6	6	0.286722	0.286722	0.286722
$\geqslant 17$	2617	56939	1	6.000	6	6	0.285408	0.285408	0.285408
$\geqslant 16$	2847	60583	1	6.000	6	6	0.294509	0.294509	0.294509
$\geqslant 15$	3105	64412	1	6.000	6	6	0.291864	0.291864	0.291864
$\geqslant 14$	3407	68613	1	5.000	5	5	0.303075	0.303075	0.303075
$\geqslant 13$	3746	72978	1	6.000	6	6	0.299836	0.299836	0.299836
$\geqslant 12$	4160	77882	1	10.000	10	10	0.292280	0.292280	0.292280
$\geqslant 11$	4634	83039	1	12.000	12	12	0.298413	0.298413	0.298413
$\geqslant 10$	5182	88462	1	10.000	10	10	0.306681	0.306681	0.306681
$\geqslant 9$	5883	94709	1	14.000	14	14	0.318836	0.318836	0.318836
$\geqslant 8$	6750	101564	1	13.000	13	13	0.318698	0.318698	0.318698
$\geqslant 7$	7904	109561	1	18.000	18	18	0.326037	0.326037	0.326037
$\geqslant 6$	9392	118389	1	17.000	17	17	0.333980	0.333980	0.333980
$\geqslant 5$	11483	128731	1	24.000	24	24	0.332774	0.332774	0.332774
$\geqslant 4$	14864	142112	1	37.000	37	37	0.339470	0.339470	0.339470
$\geqslant 3$	21812	162691	1	61.000	61	61	0.372741	0.372741	0.372741
$\geqslant 2$	41659	201678	4	252.000	252	252	0.409148	0.409148	0.409148

Table 9.15: Applying the Multilevel algorithm for community finding implemented in `igraph` by successively including vertices with lower coreness on the undirected graph induced by the assertions with positive polarity (self-loops are removed). In every row we have the number of vertices and the number of edges of each such subgraph together with the number of components (|C|) that we find in that subgraph. The next three columns present the number of communities found by the algorithm; the average among all runs, the minimum, and the maximum. The next three columns present the modularity achieved by the algorithm due to the cut induced by the communities; the average among all runs, the minimum, and the maximum. The entire computation lasted 687.9 seconds for 100 runs; that is about 6.879 seconds per run.

coreness	|V|	|E|	|C|	communities found			modularity		
				avg	min	max	avg	min	max
$\geqslant 26$	869	20526	1	5.000	5	5	0.322157	0.322157	0.322157
$\geqslant 25$	1167	27810	1	6.000	6	6	0.320322	0.320322	0.320322
$\geqslant 24$	1358	32314	1	7.000	7	7	0.320777	0.320777	0.320777
$\geqslant 23$	1514	35870	1	6.000	6	6	0.322224	0.322224	0.322224
$\geqslant 22$	1709	40099	1	7.000	7	7	0.322961	0.322961	0.322961
$\geqslant 21$	1865	43330	1	8.000	8	8	0.314372	0.314372	0.314372
$\geqslant 20$	2007	46145	1	7.000	7	7	0.320638	0.320638	0.320638
$\geqslant 19$	2173	49265	1	7.000	7	7	0.321349	0.321349	0.321349
$\geqslant 18$	2384	53011	1	8.000	8	8	0.329111	0.329111	0.329111
$\geqslant 17$	2617	56939	1	9.000	9	9	0.330618	0.330618	0.330618
$\geqslant 16$	2847	60583	1	6.000	6	6	0.331601	0.331601	0.331601
$\geqslant 15$	3105	64412	1	7.000	7	7	0.336495	0.336495	0.336495
$\geqslant 14$	3407	68613	1	9.000	9	9	0.347883	0.347883	0.347883
$\geqslant 13$	3746	72978	1	9.000	9	9	0.344480	0.344480	0.344480
$\geqslant 12$	4160	77882	1	10.000	10	10	0.349346	0.349346	0.349346
$\geqslant 11$	4634	83039	1	10.000	10	10	0.348056	0.348056	0.348056
$\geqslant 10$	5182	88462	1	10.000	10	10	0.361789	0.361789	0.361789
$\geqslant 9$	5883	94709	1	9.000	9	9	0.363861	0.363861	0.363861
$\geqslant 8$	6750	101564	1	10.000	10	10	0.368195	0.368195	0.368195
$\geqslant 7$	7904	109561	1	10.000	10	10	0.374810	0.374810	0.374810
$\geqslant 6$	9392	118389	1	10.000	10	10	0.386815	0.386815	0.386815
$\geqslant 5$	11483	128731	1	12.000	12	12	0.391540	0.391540	0.391540
$\geqslant 4$	14864	142112	1	11.000	11	11	0.401597	0.401597	0.401597
$\geqslant 3$	21812	162691	1	13.000	13	13	0.419143	0.419143	0.419143
$\geqslant 2$	41659	201678	4	25.000	25	25	0.449455	0.449455	0.449455

Table 9.16: Applying the Label Propagation algorithm for community finding implemented in `igraph` by successively including vertices with lower coreness on the induced undirected graph (self-loops are removed). In every row we have the number of vertices and the number of edges of each such subgraph together with the number of components ($|C|$) that we find in that subgraph. The next three columns present the number of communities found by the algorithm; the average among all runs, the minimum, and the maximum. The next three columns present the modularity achieved by the algorithm due to the cut induced by the communities; the average among all runs, the minimum, and the maximum. Finally the last two columns present in how many runs the algorithm computed as many communities as we had components in that subgraph. The entire computation lasted 2065.4 seconds for 100 runs; that is about 20.65 seconds per run.

coreness	$\lvert V \rvert$	$\lvert E \rvert$	$\lvert C \rvert$	communities found			modularity			agreement	
				avg	min	max	avg	min	max	Y	N
$\geqslant 26$	869	20526	1	1.000	1	1	0.000000	0.000000	0.000000	100	0
$\geqslant 25$	1167	27810	1	1.010	1	2	0.002823	0.000000	0.282317	99	1
$\geqslant 24$	1358	32314	1	1.060	1	2	0.016925	0.000000	0.284380	94	6
$\geqslant 23$	1514	35870	1	1.000	1	1	0.000000	0.000000	0.000000	100	0
$\geqslant 22$	1709	40099	1	1.010	1	2	0.002825	0.000000	0.282457	99	1
$\geqslant 21$	1865	43330	1	1.000	1	1	0.000000	0.000000	0.000000	100	0
$\geqslant 20$	2007	46145	1	1.030	1	2	0.008194	0.000000	0.275439	97	3
$\geqslant 19$	2173	49265	1	1.010	1	2	0.002720	0.000000	0.272007	99	1
$\geqslant 18$	2384	53011	1	1.010	1	2	0.002776	0.000000	0.277615	99	1
$\geqslant 17$	2617	56939	1	1.000	1	1	0.000000	0.000000	0.000000	100	0
$\geqslant 16$	2847	60583	1	1.020	1	2	0.005575	0.000000	0.278782	98	2
$\geqslant 15$	3105	64412	1	1.000	1	1	0.000000	0.000000	0.000000	100	0
$\geqslant 14$	3407	68613	1	1.010	1	2	0.002807	0.000000	0.280742	99	1
$\geqslant 13$	3746	72978	1	1.000	1	1	0.000000	0.000000	0.000000	100	0
$\geqslant 12$	4160	77882	1	1.000	1	1	0.000000	0.000000	0.000000	100	0
$\geqslant 11$	4634	83039	1	1.010	1	2	0.002830	0.000000	0.282966	99	1
$\geqslant 10$	5182	88462	1	1.000	1	1	0.000000	0.000000	0.000000	100	0
$\geqslant 9$	5883	94709	1	1.020	1	3	0.002928	0.000000	0.292798	99	1
$\geqslant 8$	6750	101564	1	1.000	1	1	0.000000	0.000000	0.000000	100	0
$\geqslant 7$	7904	109561	1	1.020	1	3	0.002960	0.000000	0.295950	99	1
$\geqslant 6$	9392	118389	1	1.000	1	1	0.000000	0.000000	0.000000	100	0
$\geqslant 5$	11483	128731	1	1.000	1	1	0.000000	0.000000	0.000000	100	0
$\geqslant 4$	14864	142112	1	1.120	1	2	0.000018	0.000000	0.000211	88	12
$\geqslant 3$	21812	162691	1	2.630	1	4	0.006172	0.000000	0.315545	5	95
$\geqslant 2$	41659	201678	4	12.710	9	19	0.009882	0.000654	0.328871	0	100

Table 9.17: Applying the InfoMAP algorithm for community finding implemented in `igraph` by successively including vertices with lower coreness on the undirected graph induced by the assertions with positive polarity (self-loops are removed). In every row we have the number of vertices and the number of edges of each such subgraph together with the number of components ($|\mathsf{C}|$) that we find in that subgraph. The next three columns present the number of communities found by the algorithm; the average among all runs, the minimum, and the maximum. The next three columns present the codelength of the partitioning found by the algorithm; the average among all runs, the minimum, and the maximum. The next three columns present the modularity achieved by the algorithm due to the cut induced by the communities; the average among all runs, the minimum, and the maximum. Finally the last two columns present in how many runs the algorithm computed as many communities as we had components in that subgraph. The entire computation lasted $13,376.27$ seconds for 10 runs; that is about $1,337.63$ seconds per run.

π	$\lvert V \rvert$	$\lvert E \rvert$	$\lvert C \rvert$	communities found			codelength			modularity			agreement	
$\geqslant$				avg	min	max	avg	min	max	avg	min	max	Y	N
26	869	20526	1	1.000	1	1	9.595067	9.595067	9.595067	0.000000	0.000000	0.000000	10	0
25	1167	27810	1	1.000	1	1	9.997333	9.997333	9.997333	0.000000	0.000000	0.000000	10	0
24	1358	32314	1	1.000	1	1	10.202537	10.202537	10.202537	0.000000	0.000000	0.000000	10	0
23	1514	35870	1	1.000	1	1	10.348016	10.348016	10.348016	0.000000	0.000000	0.000000	10	0
22	1709	40099	1	1.000	1	1	10.510817	10.510817	10.510817	0.000000	0.000000	0.000000	10	0
21	1865	43330	1	1.000	1	1	10.625889	10.625889	10.625889	0.000000	0.000000	0.000000	10	0
20	2007	46145	1	1.000	1	1	10.721326	10.721326	10.721326	0.000000	0.000000	0.000000	10	0
19	2173	49265	1	1.000	1	1	10.824262	10.824262	10.824262	0.000000	0.000000	0.000000	10	0
18	2384	53011	1	1.000	1	1	10.943376	10.943376	10.943376	0.000000	0.000000	0.000000	10	0
17	2617	56939	1	2.000	2	2	11.057997	11.057997	11.057997	0.009477	0.009477	0.009477	0	10
16	2847	60583	1	2.000	2	2	11.158651	11.158651	11.158651	0.011783	0.011783	0.011783	0	10
15	3105	64412	1	2.000	2	2	11.264781	11.264781	11.264781	0.011999	0.011999	0.011999	0	10
14	3407	68613	1	3.200	2	14	11.384844	11.374820	11.475059	0.038630	0.012607	0.272836	0	10
13	3746	72978	1	2.000	2	2	11.484748	11.484741	11.484808	0.013978	0.013944	0.014283	0	10
12	4160	77882	1	3.300	2	9	11.615842	11.607225	11.650572	0.041794	0.014052	0.153033	0	10
11	4634	83039	1	18.000	2	34	11.772474	11.733024	11.800385	0.187870	0.013987	0.338438	0	10
10	5182	88462	1	34.500	3	46	11.885121	11.859279	11.894615	0.269445	0.014270	0.345652	0	10
9	5883	94709	1	55.900	51	61	11.997908	11.994686	12.002033	0.347252	0.341475	0.354718	0	10
8	6750	101564	1	81.100	73	92	12.111098	12.107090	12.114428	0.345340	0.335496	0.356027	0	10
7	7904	109561	1	117.100	107	123	12.231296	12.228096	12.236937	0.348036	0.342610	0.353833	0	10
6	9392	118389	1	166.800	161	177	12.344162	12.338383	12.349301	0.349823	0.346514	0.353298	0	10
5	11483	128731	1	251.600	241	260	12.461265	12.456851	12.465886	0.352773	0.347126	0.357310	0	10
4	14864	142112	1	407.900	392	424	12.580570	12.575446	12.583878	0.350158	0.343223	0.355421	0	10
3	21812	162691	1	747.500	738	757	12.679398	12.673615	12.682357	0.348008	0.341139	0.353773	0	10
2	41659	201678	4	1753.500	1734	1769	12.602572	12.598905	12.604815	0.351349	0.347792	0.355575	0	10

Chapter 10

Overlapping Communities

Here we examine communities that were obtained with `CFinder` [7].

10.1 Negative Polarity

In this section we examine some communities found in the graph with negative polarity by percolating cliques.

Percolating Cliques. Table 10.1 presents how many communities were found by percolating cliques of different sizes, while Table 10.2 presents the distribution of the community sizes by percolating cliques of different sizes. Table 10.3 presents the distribution of the concepts participating in different communities by percolating cliques of different sizes. Figure 10.1 gives some examples of communities obtained by percolating cliques of size 3 and 4.

Table 10.1: Number of communities found in the undirected graph with negative polarity with `CFinder` by percolating cliques of certain size.

clique size	3	4
communities	126	24

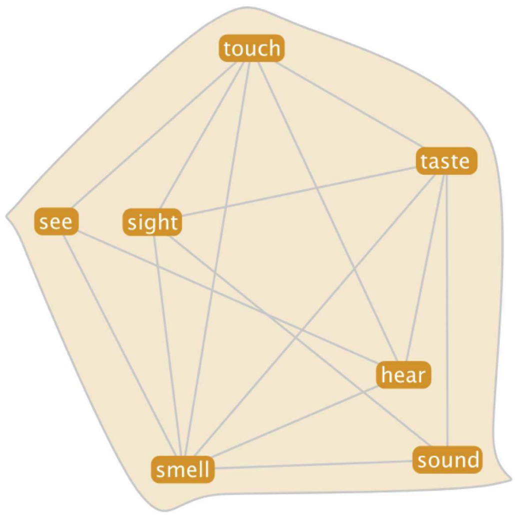
Table 10.2: Distribution of community sizes found in the undirected graph induced by the assertions of the English language with positive score and negative polarity by percolating cliques of different sizes.

percolating cliques of size	community size									
	3	4	5	6	7	9	10	11	18	457
3	75	26	14	6	2	1	–	1	–	1
4	–	16	3	2	1	–	1	–	1	–

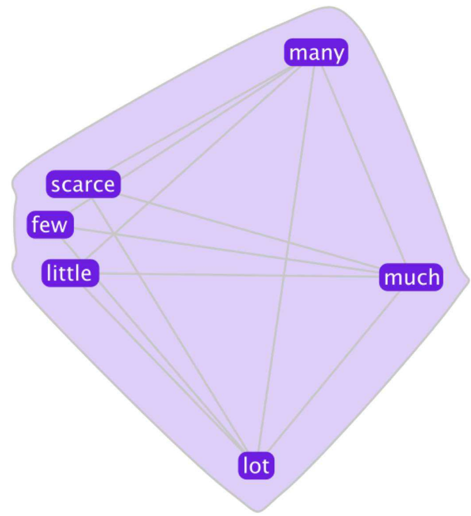
Table 10.3: Distribution of concepts participating in different communities in the undirected graph induced by the assertions of the English language with positive score and negative polarity by percolating cliques of different sizes.

percolating cliques of size	number of communities						
	0	1	2	3	4	10	13
3	10,898	716	79	10	3	1	–
4	11,603	94	8	1	–	–	1

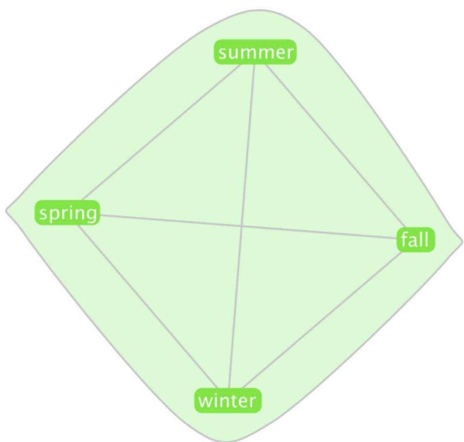
Overlapping Cliques. Overlapping cliques might prove useful in the future. They can be used for example for further clarification when posing or processing questions. We might be able to use them in order to isolate lower degree concepts related to specific questions which in turn might help by contributing in a spreading activation



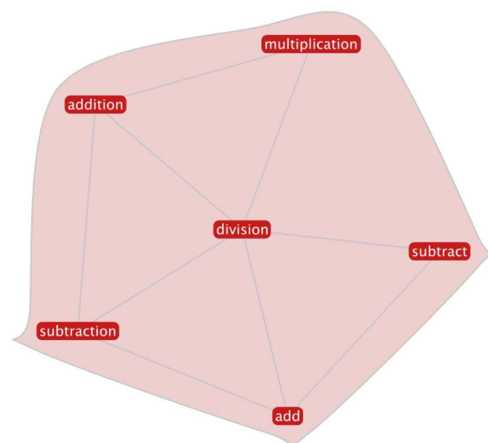
(a) Percolating senses. Seven nodes by percolating cliques of size 4. A link is missing between `sight` and `hear` generating a clique of size 5.



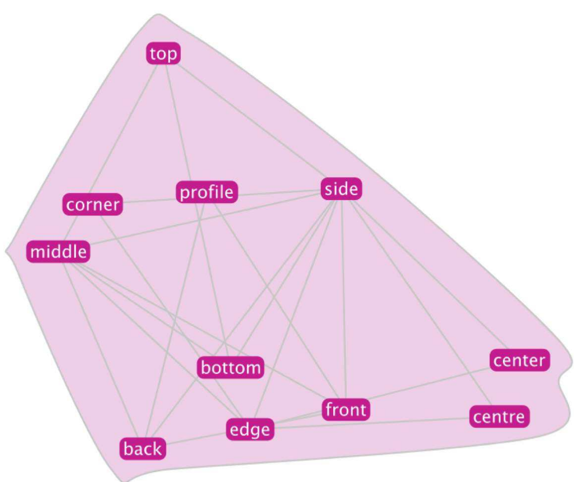
(b) Percolating frequent. Six nodes by percolating cliques of size 4.



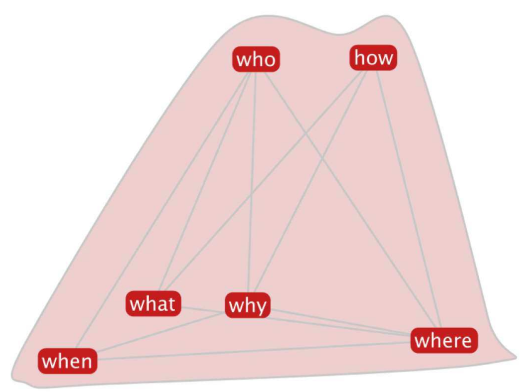
(c) Percolating year. Four nodes by percolating cliques of size 4.



(d) Percolating arithmetic. Six nodes by percolating cliques of size 3.



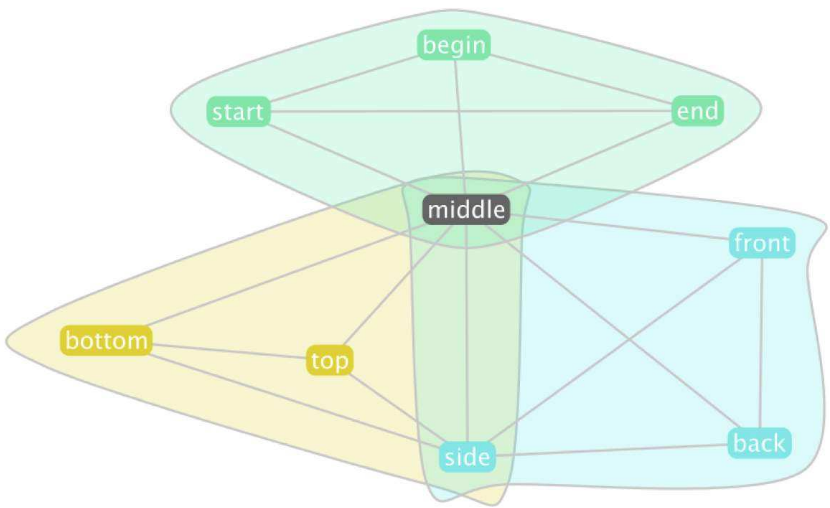
(e) Percolating orientation. Eleven nodes by percolating cliques of size 3. The concept `profile` appears here but not in Figure 5.4a.



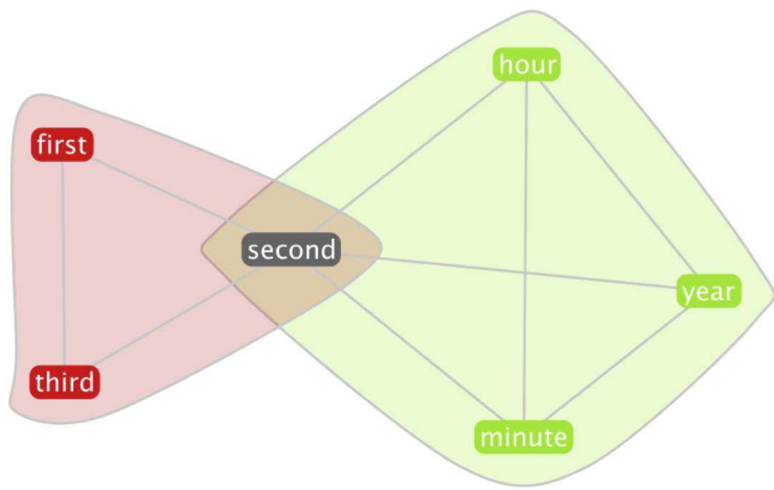
(f) Percolating questions. Six nodes by percolating cliques of size 3.

Figure 10.1: Instances of communities that are generated by percolating cliques of size 3 and 4.

process. Figures 10.2a and 10.2b give some examples of overlapping communities in the graph induced by the assertions with negative polarity (and positive score).



(a) Overlapping middle.



(b) Overlapping second.

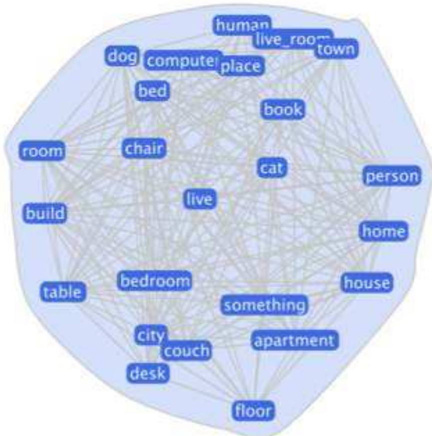
Figure 10.2: Ovelapping communities; negative polarity.

10.2 Positive Polarity

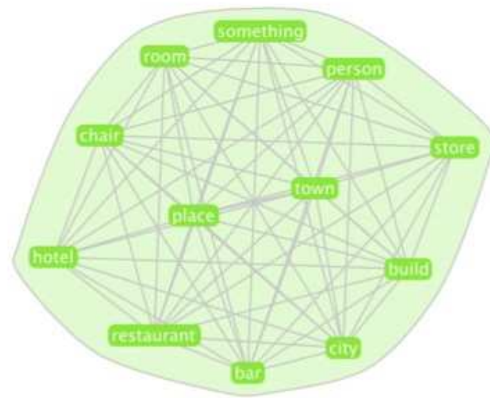
In this section we examine some communities found in the graph with positive polarity by percolating cliques.

Percolating Cliques. Table 10.4 presents how many communities were found by percolating cliques of different sizes, while Table 10.5 presents the distribution of the community sizes by percolating cliques of different sizes.

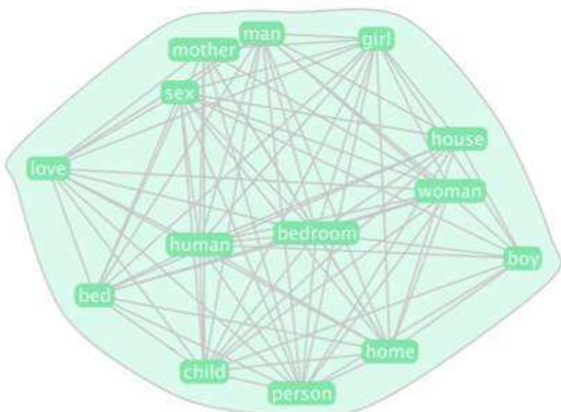
Figures 10.3 and 10.4 present communities that occur by percolating cliques of various sizes. Note that in the case of Figure 10.3c the concept `boy` does make it and is part of the community as it would be expected contrasting the fact that it does not appear in the relevant clique of size 11 shown in Table 8.3. As another example, one would also expect the concept `dishonest` or `dishonesty` to appear in the community shown in Figure 10.4d. Moreover, through percolation we can get hints about missing edges.



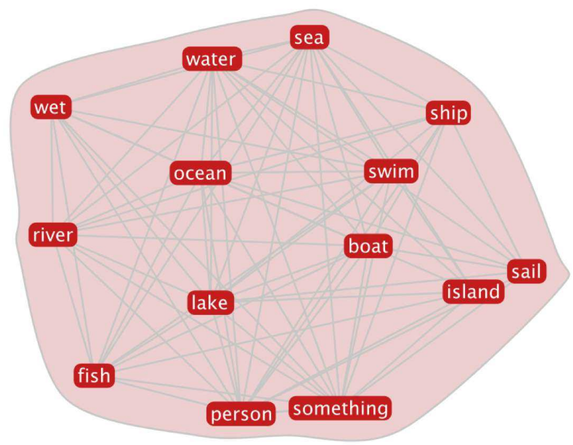
(a) Percolating house. Twenty four nodes by percolating cliques of size 11.



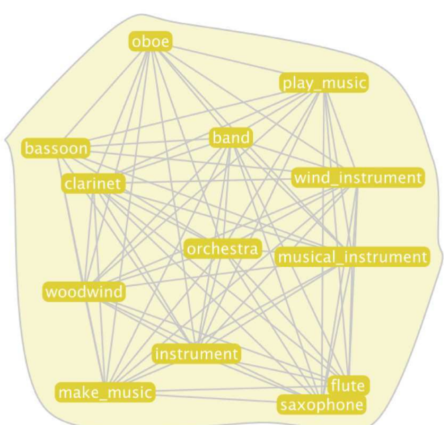
(b) Percolating neighborhood. Twelve nodes by percolating cliques of size 11.



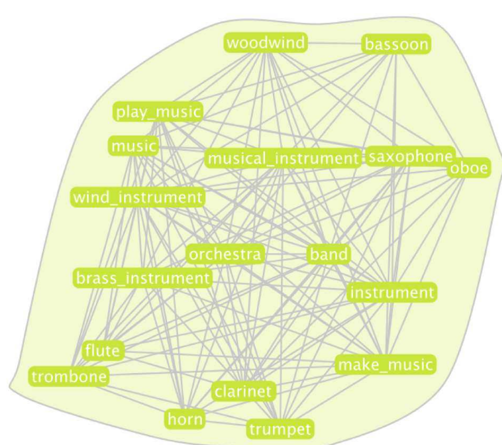
(c) Percolating human reproduction. Fourteen nodes by percolating cliques of size 10.



(d) Percolating sea. Fourteen nodes by percolating cliques of size 9.



(e) Percolating music. Fourteen nodes by percolating cliques of size 9.



(f) Percolating music. Fourteen nodes by percolating cliques of size 8.

Figure 10.3: Percolating cliques of sizes 8, 9, 10, and 11 and some interesting communities.

Table 10.4: Number of communities found in the undirected graph with positive polarity with `CFinder` by percolating cliques of certain size.

size	3	4	5	6	7	8	9	10	11	12
comm.	362	290	287	209	120	84	16	12	6	1

Nested Clique. We can observe *nested* cliques in `ConceptNet 4`. One such instance appears by percolating cliques of size 9 and is shown in Figure 10.5. The community shown in Figure 10.5a is composed of 128 concepts, while Figure 10.5b presents a community composed by a clique of size 9 which does not percolate to include more concepts. In the big clique we can see the concepts appearing in the smaller clique either on the lower right hand side, or in the middle.

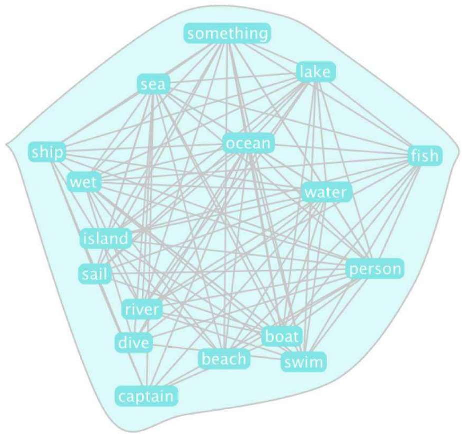
Overlapping Cliques. Figures 10.6a, 10.6b, and 10.7b give some examples of overlapping communities in the graph induced by the assertions with positive polarity (and positive score).

Table 10.5: Distribution of community sizes found in the undirected graph induced by the assertions of the English language with positive score and positive polarity by percolating cliques of different sizes.

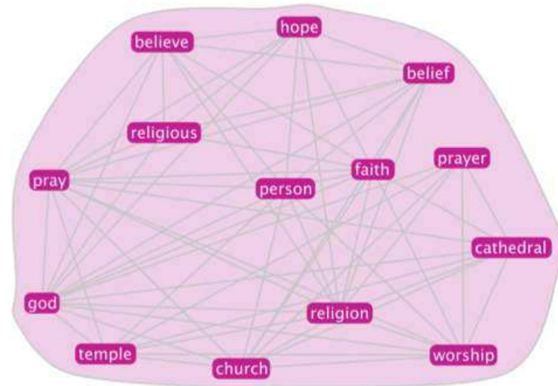
community size	percolating cliques of size									
	3	4	5	6	7	8	9	10	11	12
3	320	–	–	–	–	–	–	–	–	–
4	31	236	–	–	–	–	–	–	–	–
5	8	35	204	–	–	–	–	–	–	–
6	1	10	43	122	–	–	–	–	–	–
7	–	3	18	41	61	–	–	–	–	–
8	1	2	9	21	22	38	–	–	–	–
9	–	1	7	7	9	19	6	–	–	–
10	–	–	1	5	5	8	2	3	–	–
11	–	–	–	4	5	5	1	5	3	–
12	–	1	1	3	3	3	1	1	1	1
13	–	–	1	–	2	1	2	–	–	–
14	–	–	–	3	5	4	1	1	1	–
15	–	–	–	1	–	–	–	1	–	–
16	–	–	–	–	2	1	–	–	–	–
17	–	–	1	–	–	1	1	–	–	–
18	–	–	1	–	–	2	1	–	–	–
21	–	1	–	–	–	–	–	–	–	–
22	–	–	–	–	1	–	–	–	–	–
23	–	–	–	–	1	–	–	–	–	–
24	–	–	–	1	–	–	–	–	1	–
25	–	–	–	–	1	–	–	–	–	–
37	–	–	–	–	–	1	–	–	–	–
47	–	–	–	–	1	–	–	–	–	–
49	–	–	–	–	1	–	–	1	–	–
128	–	–	–	–	–	–	1	–	–	–
278	–	–	–	–	–	1	–	–	–	–
796	–	–	–	–	1	–	–	–	–	–
1944	–	–	–	1	–	–	–	–	–	–
3868	–	–	1	–	–	–	–	–	–	–
8208	–	1	–	–	–	–	–	–	–	–
22533	1	–	–	–	–	–	–	–	–	–

Table 10.6: Distribution of concepts participating in different communities in the undirected graph induced by the assertions of the English language with positive score and positive polarity by percolating cliques of different sizes.

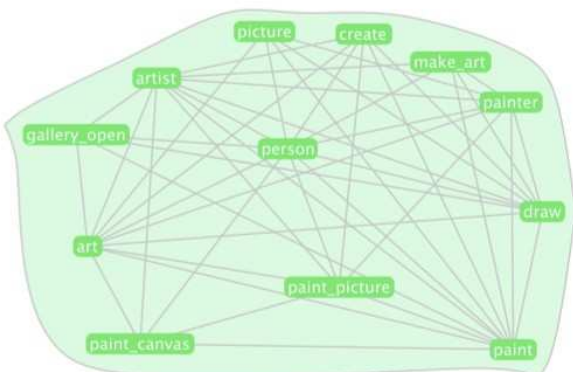
number of communities	percolating cliques of size									
	3	4	5	6	7	8	9	10	11	12
0	233,812	248,357	252,646	254,592	255,701	256,285	256,629	256,760	256,810	256,834
1	22,451	7,727	3,433	1,691	805	361	162	50	16	12
2	547	629	526	326	197	90	43	13	7	–
3	29	94	159	140	70	59	5	12	4	–
4	4	30	46	51	25	13	5	3	5	–
5	–	5	13	22	21	12	–	3	3	–
6	1	2	10	7	6	10	–	2	1	–
7	1	–	4	5	5	3	–	2	–	–
8	–	1	1	3	5	6	1	–	–	–
9	–	–	3	–	4	1	1	–	–	–
10	1	–	1	–	1	–	–	1	–	–
11	–	–	–	2	2	–	–	–	–	–
12	–	–	1	1	2	1	–	–	–	–
13	–	–	1	2	–	–	–	–	–	–
14	–	–	–	–	–	1	–	–	–	–
16	–	–	–	–	–	1	–	–	–	–
18	–	–	–	1	–	1	–	–	–	–
19	–	–	–	1	–	1	–	–	–	–
21	–	–	1	–	–	–	–	–	–	–
24	–	–	–	–	1	–	–	–	–	–
25	–	–	–	1	–	–	–	–	–	–
34	–	1	–	–	–	–	–	–	–	–
52	–	–	–	–	–	1	–	–	–	–
74	–	–	–	–	1	–	–	–	–	–
87	–	–	1	–	–	–	–	–	–	–
105	–	–	–	1	–	–	–	–	–	–



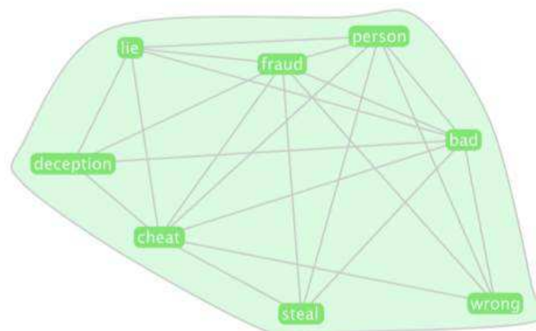
(a) Percolating water. Fourteen nodes by percolating cliques of size 8.



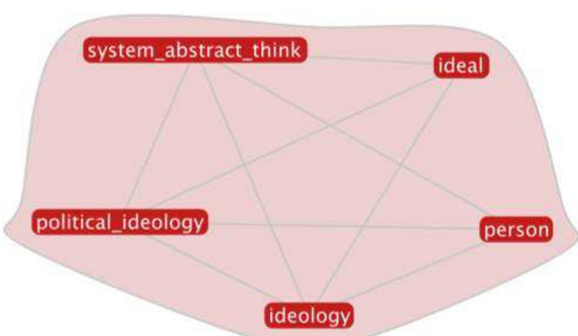
(b) Percolating religion. Fourteen nodes by percolating cliques of size 7.



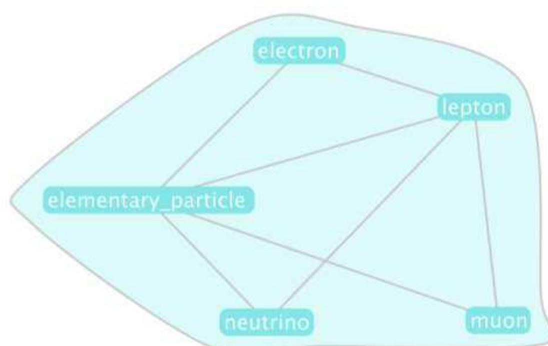
(c) Percolating painter. Twelve nodes by percolating cliques of size 6.



(d) Percolating dishonest/dishonesty. Eight nodes by percolating cliques of size 5. Note that `dishonest/dishonesty` is missing from the community.

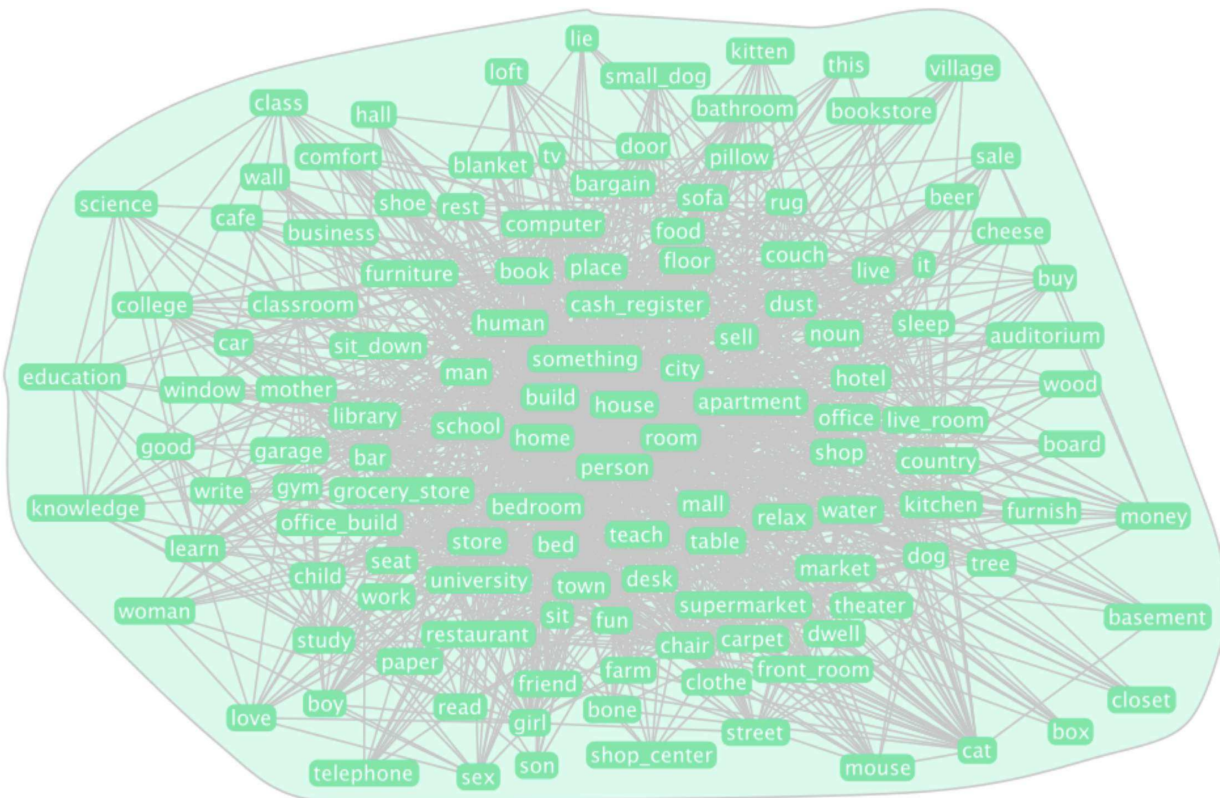


(e) Percolating ideals. Five nodes by percolating cliques of size 4.

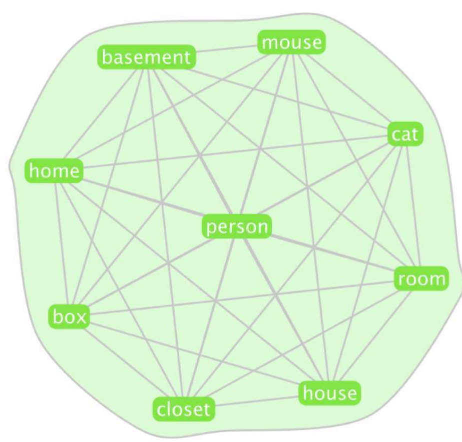


(f) Percolating particles. Five nodes by percolating cliques of size 3.

Figure 10.4: Percolating cliques of sizes 3, 4, 5, 6, 7 and 8 and one interesting community in each case.

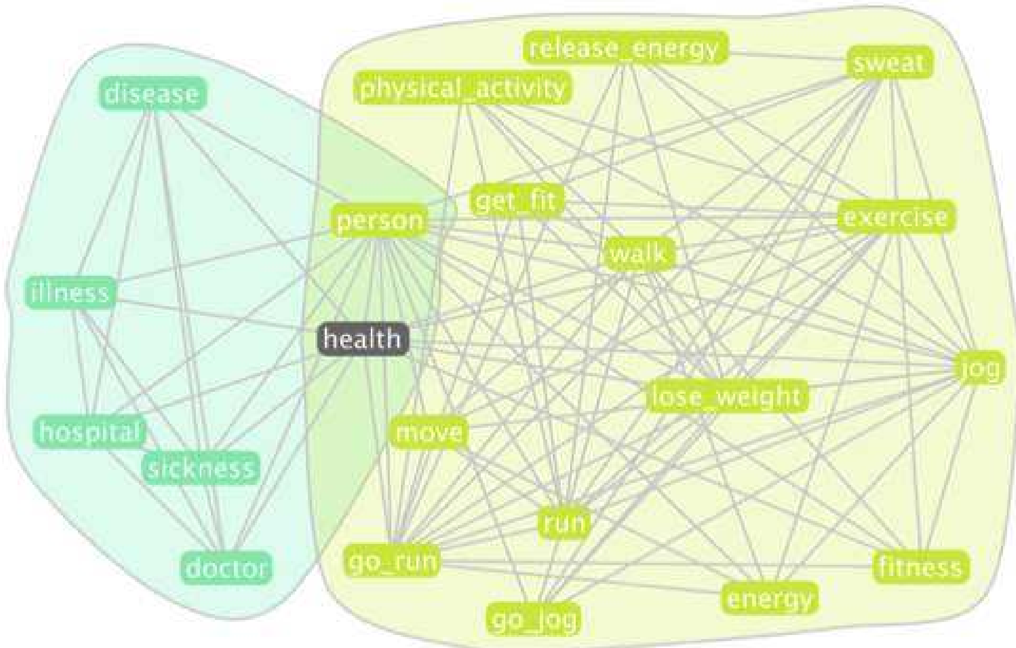


(a) A big community composed of 128 concepts by percolating cliques of size 9.

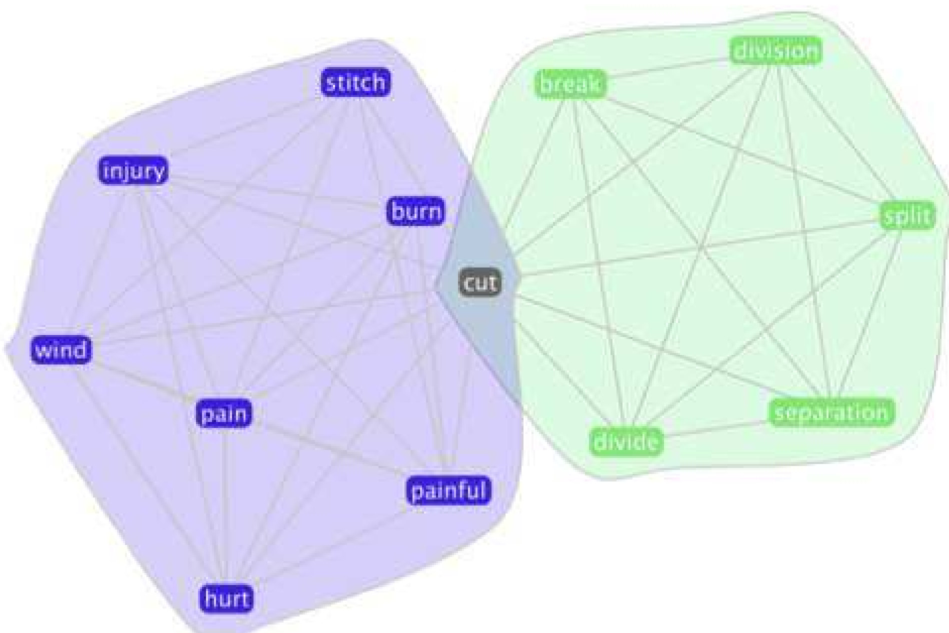


(b) A community composed of 9 concepts that form a clique which does not percolate to include more concepts.

Figure 10.5: Nested cliques. A clique of size 9 has concepts which appear in a bigger clique.

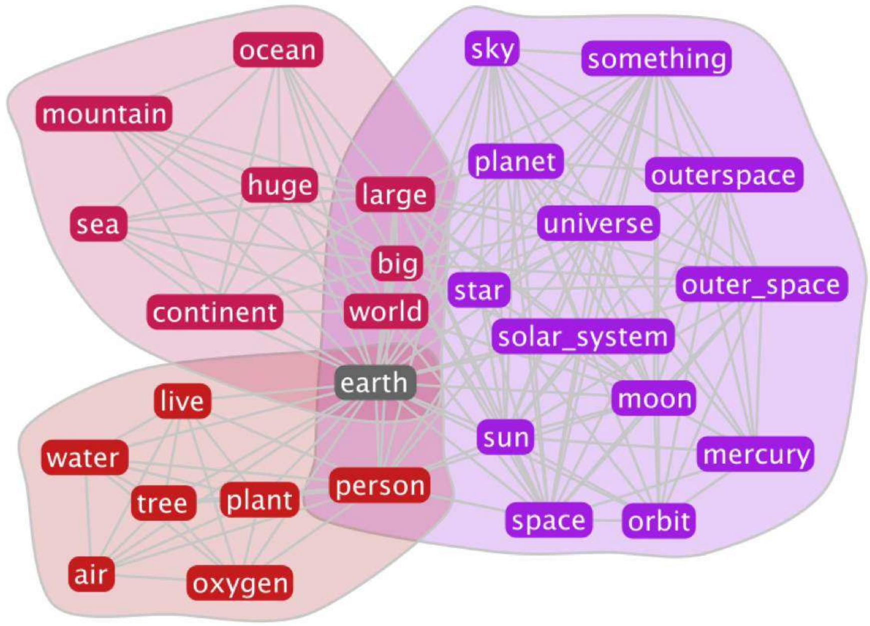


(a) Overlapping health.

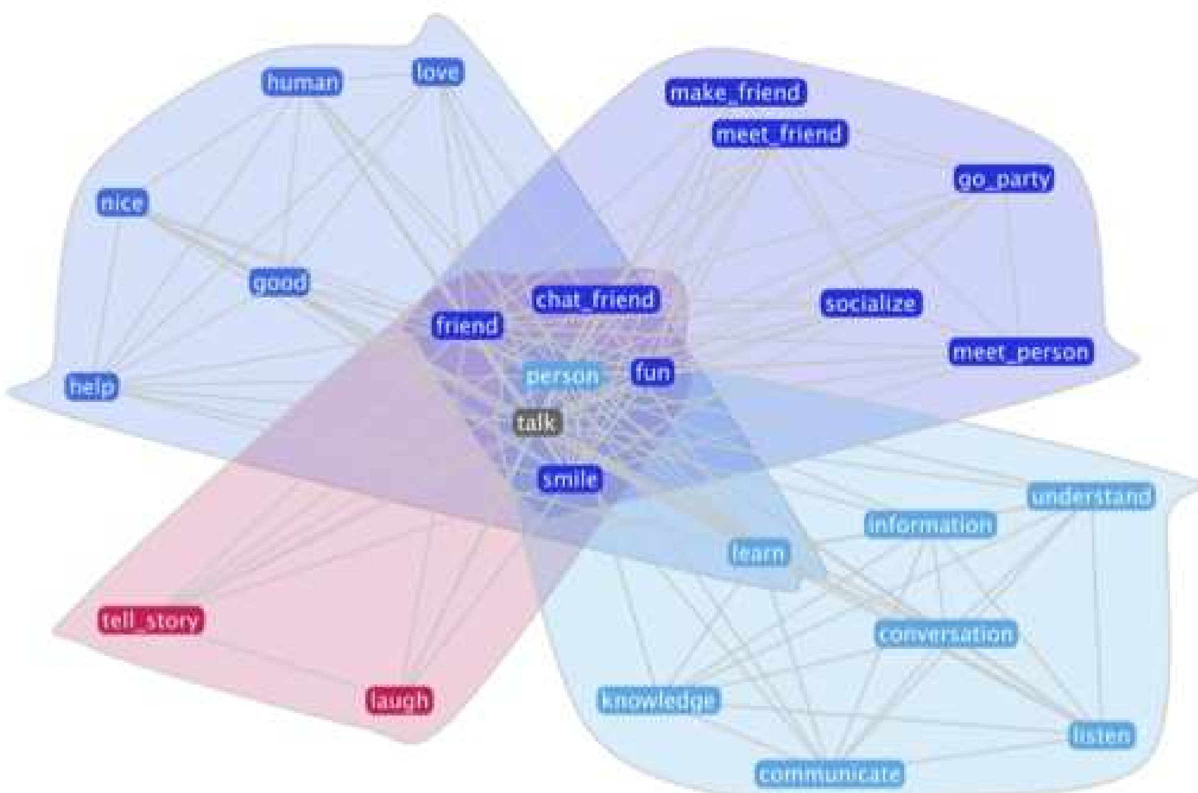


(b) Overlapping cut.

Figure 10.6: Concepts participating in more than one communities.



(a) Overlapping earth.



(b) Overlapping talk.

Figure 10.7: Concepts participating in more than one communities.

Part IV

Mining

Chapter 11

Mining Rules

In this chapter we discuss the application of data mining towards the automated construction of a background theory for the relations used in the knowledge base. We consider rules of the simplest form, mainly for computational considerations.

A *rule* is given by an ordered triple of relations (`X`, `Y`, `Z`), where `X`, `Y` are the *premisses* and `Z` is the *conclusion*. For such a triple we consider triples of concepts (`a`, `b`, `c`) such that the assertions

(`a`, `X`, `b`) and (`b`, `Y`, `c`)

are in the knowledge base. Such triples form the *support* of the rule. If (`a`, `Z`, `c`) is also in the knowledge base then (`a`, `b`, `c`) is a *success* for the rule (`X`, `Y`, `Z`), otherwise it is a *failure*. The *success rate* of a rule is the percentage of successes in the support. Consider, for example, the rule (`Desires`, `LocatedNear`, `AtLocation`) and the triple of concepts (`human`, `drink`, `bar`). The assertions (`human`, `Desires`, `drink`) and (`drink`, `LocatedNear`, `bar`) are both in the knowledge base. Therefore, we check whether the assertion (`human`, `AtLocation`, `bar`) is in the knowledge base. It is, so (`human`, `drink`, `bar`) is a success for the rule (`Desires`, `LocatedNear`, `AtLocation`).

A triple of concepts (`a`, `b`, `c`) is *valid* for a rule (`X`, `Y`, `Z`) if the claim

(`a`, `X`, `b`) and (`b`, `Y`, `c`) therefore (`a`, `Z`, `c`)

makes sense as a reasoning step. Otherwise (`a`, `b`, `c`) is *invalid*. *Making sense is a subjective judgement* and its intended meaning is up for discussion. In what follows we use the sense "given that the premisses hold it is reasonable to assume that the conclusion holds". For example, (`human`, `drink`, `bar`) is valid for the rule (`Desires`, `LocatedNear`, `AtLocation`). Note that by the nature of its definition, deciding about validity requires an (often ambiguous) decision by a human and so computing precise statistics about it is difficult.

We performed an exhaustive test for all possible rules involving relations that have at least 300 assertions with positive score regardless of their polarity. We searched for *frequent* rules, with support at least 300 and success rate at least 5% [1]. Success rates are expected to be low even for correct rules due to the sparsity of the network. Tables 11.1 and 11.2 present the 76 triples of relations that satisfy these conditions; that is, at least 300 assertions are in the support and at least 5% success rate. Below we give examples of some such relations, plus an interesting one with low success rate, and comment on issues raised by these examples.

Our first example is the rule (`Desires`, `LocatedNear`, `AtLocation`). This is the highest scoring rule with 251 successes and support 2050 (12% success rate). The triples (`human`, `drink`, `bar`) and (`bird`, `seed`, `garden`) are successful and valid. The triple (`human`, `love`, `heart`) is successful but invalid. The triple (`bird`, `seed`, `plant garden`) is a failure but it is valid. The reason for the failure is that the assertion (`bird`, `AtLocation`, `plant garden`) is missing from the knowledge base. This is an example of using the mined rules to identify missing entries.

The rule (`AtLocation`, `PartOf`, `AtLocation`) has 2,394 successes and support 27,917 (8.5% success rate). The triple (`text book`, `classroom`, `school`) is successful and valid. On the other hand, (`text book`, `classroom`, `school system`) is a failure. In contrast to the failure discussed for the first rule above, this is not due to a missing assertion, because the triple is *invalid*. This points to a general problem with this rule: it is only expected

[1] For rules involving more than three concepts such an exhaustive search is not feasible, and it will be necessary to use more advanced data mining techniques.

to hold if the third concept is a physical object, like `school` and unlike `school system`. Thus examining this example suggests a weakening of the rule.

The rule (`PartOf`, `AtLocation`, `AtLocation`) is similar to the previous one. However, its success rate is much smaller, only 1.4% (with support 78,804, but only 1,112 successes). A possible explanation of the discrepancy can be illustrated by the triple (`engine oil`, `car`, `town`). It is a failure as the assertion (`engine oil`, `AtLocation`, `town`) is not in the knowledge base. Its validity depends on the status of (`engine oil`, `AtLocation`, `town`). This assertion is not to be expected as input from a user (or from a text). On the other hand, it is reasonable as a factual statement about the world.

Let us elaborate on the difference between the two rules. For (`AtLocation`, `PartOf`, `AtLocation`), the combined facts that `a` is an appropriate[2] left argument for `AtLocation`, `b` is an appropriate right argument for `AtLocation`, and (`b`, `PartOf`, `c`) mean that if `c` is an appropriate right argument for `AtLocation` (like `school` but unlike `school system`) then the assertion (`a`, `AtLocation`, `c`) *makes sense both as a factual statement about the world and in terms of natural language usage*. By way of contrast, for (`PartOf`, `AtLocation`, `AtLocation`), things that are appropriate as left arguments for `PartOf` are normally not thought of as appropriate left arguments for `AtLocation`; if they do occur as such a left argument then they occur as being `AtLocation` of the thing they are part of. Thus, in this case (`a`, `AtLocation`, `c`) *may make sense as a factual statement about the world but not in terms of natural language usage*. Thus, the observed difference between the success rates of two similar rules points to a *possible mismatch between natural language usage and intended question answering applications*. This may be an issue to consider for further knowledge base development.

The rule (`LocatedNear`, `PartOf`, `IsA`) does not make much sense even if it has 253 successes and support $4,252$ (6% success rate). Most successes we examined are false or nonsensical. This is an example of a rule with high success rate but with many successful, invalid triples. An example is the triple (`desk`, `classroom`, `school`). The wrong assertion (`desk`, `IsA`, `school`) comes from the sentence `Schools have desks` through the intermediate form `Desk is a type of school`. Thus the problem presumably comes from a programming error and fixing it might eliminate many wrong assertions. Hence this in an example where rule mining can be used to correct mistakes.

[2]By *appropriate* we mean "makes common sense for users asked to give natural language statements".

Table 11.1: The first 37 out of the 76 triples that appear to have support of at least 300 and success rate at least 5%. Note that we have neglected from the computation all those relations that do not have at least 300 assertions with positive score regardless of their polarity.

relation X	relation Y	relation Z	ratio	successes	support
HasFirstSubevent (1)	MadeOf (4)	Causes (18)	0.058282	19	326
AtLocation (6)	AtLocation (6)	AtLocation (6)	0.053967	29053	538349
AtLocation (6)	PartOf (21)	AtLocation (6)	0.085754	2394	27917
AtLocation (6)	LocatedNear (30)	AtLocation (6)	0.073569	4942	67175
AtLocation (6)	SimilarSize (31)	AtLocation (6)	0.082580	1919	23238
CapableOf (8)	AtLocation (6)	AtLocation (6)	0.115059	3067	26656
CapableOf (8)	CausesDesire (17)	CapableOf (8)	0.105263	424	4028
CapableOf (8)	LocatedNear (30)	AtLocation (6)	0.084516	530	6271
Desires (10)	AtLocation (6)	AtLocation (6)	0.109640	2707	24690
Desires (10)	CausesDesire (17)	CapableOf (8)	0.117987	286	2424
Desires (10)	PartOf (21)	AtLocation (6)	0.055444	55	992
Desires (10)	CreatedBy (25)	IsA (5)	0.055233	19	344
Desires (10)	CreatedBy (25)	AtLocation (6)	0.055233	19	344
Desires (10)	CreatedBy (25)	CapableOf (8)	0.055233	19	344
Desires (10)	LocatedNear (30)	AtLocation (6)	0.122439	251	2050
Desires (10)	LocatedNear (30)	Desires (10)	0.053659	110	2050
Desires (10)	SimilarSize (31)	AtLocation (6)	0.058719	33	562
ConceptuallyRelatedTo (12)	ConceptuallyRelatedTo (12)	IsA (5)	0.052977	7052	133114
ConceptuallyRelatedTo (12)	ConceptuallyRelatedTo (12)	HasProperty (20)	0.054194	7214	133114
ConceptuallyRelatedTo (12)	PartOf (21)	AtLocation (6)	0.059574	579	9719
ConceptuallyRelatedTo (12)	PartOf (21)	ConceptuallyRelatedTo (12)	0.051960	505	9719
ConceptuallyRelatedTo (12)	PartOf (21)	HasProperty (20)	0.051549	501	9719
ConceptuallyRelatedTo (12)	LocatedNear (30)	IsA (5)	0.060326	1611	26705
ConceptuallyRelatedTo (12)	LocatedNear (30)	AtLocation (6)	0.062872	1679	26705
ConceptuallyRelatedTo (12)	LocatedNear (30)	ConceptuallyRelatedTo (12)	0.065007	1736	26705
ConceptuallyRelatedTo (12)	LocatedNear (30)	HasProperty (20)	0.063471	1695	26705
ConceptuallyRelatedTo (12)	SimilarSize (31)	IsA (5)	0.067656	629	9297
ConceptuallyRelatedTo (12)	SimilarSize (31)	ConceptuallyRelatedTo (12)	0.068409	636	9297
ConceptuallyRelatedTo (12)	SimilarSize (31)	HasProperty (20)	0.065182	606	9297
HasA (16)	PartOf (21)	IsA (5)	0.072956	706	9677
HasA (16)	PartOf (21)	AtLocation (6)	0.050842	492	9677
HasA (16)	PartOf (21)	ConceptuallyRelatedTo (12)	0.059729	578	9677
HasA (16)	PartOf (21)	HasProperty (20)	0.059419	575	9677
HasA (16)	LocatedNear (30)	IsA (5)	0.060117	1233	20510
HasA (16)	LocatedNear (30)	AtLocation (6)	0.052365	1074	20510
HasA (16)	LocatedNear (30)	ConceptuallyRelatedTo (12)	0.060751	1246	20510
HasA (16)	LocatedNear (30)	HasProperty (20)	0.059191	1214	20510

Table 11.2: The last 39 out of the 76 triples that appear to have support of at least 300 and success rate at least 5%. Note that we have neglected from the computation all those relations that do not have at least 300 assertions with positive score regardless of their polarity.

relation X	relation Y	relation Z	ratio	successes	support
HasProperty (20)	LocatedNear (30)	AtLocation (6)	0.051700	1743	33714
HasProperty (20)	LocatedNear (30)	HasProperty (20)	0.053183	1793	33714
HasProperty (20)	SimilarSize (31)	IsA (5)	0.057224	642	11219
HasProperty (20)	SimilarSize (31)	ConceptuallyRelatedTo (12)	0.055263	620	11219
HasProperty (20)	SimilarSize (31)	HasProperty (20)	0.063909	717	11219
CreatedBy (25)	LocatedNear (30)	AtLocation (6)	0.058252	42	721
LocatedNear (30)	ConceptuallyRelatedTo (12)	HasProperty (20)	0.051206	2416	47182
LocatedNear (30)	PartOf (21)	IsA (5)	0.059501	253	4252
LocatedNear (30)	PartOf (21)	AtLocation (6)	0.097601	415	4252
LocatedNear (30)	PartOf (21)	ConceptuallyRelatedTo (12)	0.063735	271	4252
LocatedNear (30)	PartOf (21)	HasProperty (20)	0.070790	301	4252
LocatedNear (30)	PartOf (21)	PartOf (21)	0.062088	264	4252
LocatedNear (30)	LocatedNear (30)	IsA (5)	0.062558	738	11797
LocatedNear (30)	LocatedNear (30)	AtLocation (6)	0.080698	952	11797
LocatedNear (30)	LocatedNear (30)	ConceptuallyRelatedTo (12)	0.068407	807	11797
LocatedNear (30)	LocatedNear (30)	HasProperty (20)	0.070018	826	11797
LocatedNear (30)	LocatedNear (30)	LocatedNear (30)	0.058913	695	11797
LocatedNear (30)	SimilarSize (31)	IsA (5)	0.051922	204	3929
LocatedNear (30)	SimilarSize (31)	AtLocation (6)	0.062611	246	3929
LocatedNear (30)	SimilarSize (31)	ConceptuallyRelatedTo (12)	0.059812	235	3929
LocatedNear (30)	SimilarSize (31)	HasProperty (20)	0.052940	208	3929
SimilarSize (31)	MadeOf (4)	HasA (16)	0.063559	60	944
SimilarSize (31)	MadeOf (4)	HasProperty (20)	0.050847	48	944
SimilarSize (31)	ConceptuallyRelatedTo (12)	IsA (5)	0.067418	933	13839
SimilarSize (31)	ConceptuallyRelatedTo (12)	ConceptuallyRelatedTo (12)	0.070020	969	13839
SimilarSize (31)	ConceptuallyRelatedTo (12)	HasProperty (20)	0.070742	979	13839
SimilarSize (31)	PartOf (21)	IsA (5)	0.060452	83	1373
SimilarSize (31)	PartOf (21)	AtLocation (6)	0.072833	100	1373
SimilarSize (31)	PartOf (21)	ConceptuallyRelatedTo (12)	0.065550	90	1373
SimilarSize (31)	PartOf (21)	HasProperty (20)	0.067007	92	1373
SimilarSize (31)	CreatedBy (25)	ConceptuallyRelatedTo (12)	0.055838	22	394
SimilarSize (31)	LocatedNear (30)	IsA (5)	0.056172	162	2884
SimilarSize (31)	LocatedNear (30)	AtLocation (6)	0.074202	214	2884
SimilarSize (31)	LocatedNear (30)	ConceptuallyRelatedTo (12)	0.065534	189	2884
SimilarSize (31)	LocatedNear (30)	HasProperty (20)	0.057559	166	2884
SimilarSize (31)	SimilarSize (31)	IsA (5)	0.080597	108	1340
SimilarSize (31)	SimilarSize (31)	ConceptuallyRelatedTo (12)	0.092537	124	1340
SimilarSize (31)	SimilarSize (31)	HasProperty (20)	0.088060	118	1340
SimilarSize (31)	SimilarSize (31)	SimilarSize (31)	0.053731	72	1340

Bibliography

[1] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008+, Jul 2008. Also available at arXiv:0803.0476 [physics.soc-ph].

[2] Aaron Clauset, M. E. J. Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical Review E*, 70(6):066111+, Dec 2004. Also available at arXiv:cond-mat/0408187 [cond-mat.stat-mech].

[3] Aaron Clauset, Cosma Rohilla Shalizi, and M. E. J. Newman. Power-Law Distributions in Empirical Data. *SIAM Review*, 51(4):661–703, November 2009.

[4] Gábor Csárdi and Tamás Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006.

[5] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

[6] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74:036104, Sep 2006. Also available at arXiv:physics/0605087 [physics.data-an].

[7] Gergely Palla, Imre Dernyi, Ills Farkas, and Tams Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, June 2005.

[8] Pascal Pons and Matthieu Latapy. Computing Communities in Large Networks Using Random Walks. *Journal of Graph Algorithms and Applications*, 10(2):191–218, 2006. The long version is available at arXiv:physics/0512106 [physics.soc-ph].

[9] Usha N. Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 76(3):036106+, Sep 2007. Also available at arXiv:0709.2938 [physics.soc-ph].

[10] Jörg Reichardt and Stefan Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74:016110, Jul 2006. Also available at arXiv:cond-mat/0603718 [cond-mat.dis-nn].

[11] M. Rosvall, D. Axelsson, and C. T. Bergstrom. The map equation. *The European Physical Journal Special Topics*, 178:13–23, 2009. See also arXiv:0906.1405 [physics.soc-ph].

[12] Martin Rosvall and Carl T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4):1118–1123, 2008. Also available at arXiv:0707.0609 [physics.soc-ph].

[13] Robert Speer, Catherine Havasi, and Henry Lieberman. AnalogySpace: Reducing the Dimensionality of Common Sense Knowledge. In *AAAI*, pages 548–553, 2008.

[14] Ken Wakita and Toshiyuki Tsurumi. Finding Community Structure in Mega-scale Social Networks. *CoRR*, abs/cs/0702048, 2007. Available at arXiv:cs/0702048 [cs.CY].

[15] S. Wasserman and K. Faust. *Social Network Analysis: methods and applications*. Cambridge University Press, 1994.

[16] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, Jun 1998.

Appendix A

Tables and Files in CSV Format

Here we have a brief presentation of the CSV files.

conceptnet_assertion

This is the main table of the database. Table A.1 presents the first two lines.

conceptnet_concept

This table has information related to concepts. Table A.2 presents the first two lines.

conceptnet_relation

This table describes the relations that are used to form assertions. Table A.3 presents the first two lines.

nl_frequency

Table A.4 describes the frequencies that are used in order to classify the extent to which a relation holds between two concepts in the assertions. It ranges from *never* (*polarity* is -1) to *always* (*polarity* is $+1$).

conceptnet_frame

This table has information related to frames. Table A.5 presents the first two lines.

conceptnet_surfaceform

This table has the information related to surface forms. Table A.6 presents the first two lines.

conceptnet_rawassertion

This table has information related to raw assertions. Table A.7 presents the first two lines.

corpus_sentence

This table essentially has the actual sentence to which a raw assertion points to. Table A.8 presents the first two lines.

Table A.1: The beginning of conceptnet_assertion.

id	language_id	relation_id	concept1_id	concept2_id	score	frequency_id	best_surface1_id	best surface2_id	best_raw_id	best_frame_id
2	en	6	5	6	1	1	5	6	3	3
3	en	7	7	8	1	1	7	8	4	4
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Table A.2: The beginning of conceptnet_concept.

id	language_id	text	num_assertions	words	visible
5	en	something	2887	1	1
6	en	to	71	1	1
⋮	⋮	⋮	⋮	⋮	⋮

Table A.3: The beginning of conceptnet_relation.

id	name	description
1	HasFirstSubevent	What do you do first to accomplish it?
2	HasLastSubevent	What do you do last to accomplish it?
⋮	⋮	⋮

Table A.4: The beginning of nl_frequency.

id	language_id	text	value
1	en		5
2	en	often	6
⋮	⋮	⋮	⋮

Table A.5: The beginning of conceptnet_frame.

id	language_id	text	relation_id	goodness	frequency_id	question_yn	question1	question2
3	en	Somewhere {1} can be is next {2}	6	1	1			
4	en	You can use {1} to {2}	7	2	1			
⋮	⋮	⋮	⋮	⋮	⋮			

Table A.6: The beginning of conceptnet_surfaceform.

id	language_id	concept_id	text	residue	use_count
5	en	5	something	1ly	3979
6	en	6	to	1	59
⋮	⋮	⋮	⋮	⋮	⋮

Table A.7: The beginning of conceptnet_rawassertion.

id	created	updated	sentence_id	assertion_id	creator_id	surface1_id	surface2_id	frame_id	batch_id	language_id	score
3	2009...	2009...	715991	2	997	5	6	3		en	1
4	2009...	2009...	715993	3	992	7	8	4		en	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮		⋮	⋮

Table A.8: The beginning of corpus_sentence.

id	text	creator_id	created_on	language_id	activity_id	score
715991	Somewhere something can be is next to	997	2006...	en	27	1
715992	picture description: an old house made of brick	1002	2006...	en	27	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮

A.1 Database Entries: Tables with Relations and Frequencies

Relations. `ConceptNet 4` has 30 relations; 27 appear among the assertions in the English language. Table A.9 gives an overview of all the relations found in `ConceptNet 4`.

Frequencies. Table A.10 presents the different frequencies that we can encounter in `ConceptNet 4` in the assertions of the English language.

Table A.9: The relations that we can find in `ConceptNet 4`. Note that three of them do not appear among assertions in the English language and in these cases the index assigned to them is ✗.

index	id	name	description
0	1	HasFirstSubevent	What do you do first to accomplish it?
1	2	HasLastSubevent	What do you do last to accomplish it?
2	3	HasPrerequisite	What do you need to do first?
3	4	MadeOf	What is it made of?
4	5	IsA	What kind of thing is it?
5	6	AtLocation	Where would you find it?
6	7	UsedFor	What do you use it for?
7	8	CapableOf	What can it do?
8	9	MotivatedByGoal	Why would you do it?
9	10	Desires	What does it want?
✗	11	[deprecated 1]	
10	12	ConceptuallyRelatedTo	
11	13	DefinedAs	How do you define it?
12	14	InstanceOf	*What type of thing is it a specific example of?
13	15	SymbolOf	
14	16	HasA	
15	17	CausesDesire	What does it make you want to do?
16	18	Causes	What does it make happen?
17	19	HasSubevent	What do you do to accomplish it?
18	20	HasProperty	What properties does it have?
19	21	PartOf	What is it part of?
20	22	ReceivesAction	What can you do to it?
✗	23	ObstructedBy	
21	24	InheritsFrom	
22	25	CreatedBy	How do you bring it into existence?
✗	26	Translation	
23	28	HasPainCharacter	*What is the character of pain associated with it?
24	29	HasPainIntensity	*What is the intensity of pain associated with it?
25	30	LocatedNear	
26	31	SimilarSize	

Table A.10: The different frequencies that we can encounter in the table `nl_frequency` in the English language.

index	id	text	value
0	1		5
1	2	often	7
2	3	(UNSPECIFIED)	5
3	11	always	10
4	21	never	-10
5	22	rarely	-2
6	25	not	-5
7	1209	sometimes	4
8	1215	usually	8
9	1368	occasionally	2
10	1403	almost always	9

Appendix B

Derived Input Files

In this section we present the format and properties of the input files that we are going to use.

B.1 Special Indices

Throughout the files we have reserved two special *indices* that may appear in the input files. These are the following:

Null Index = -1. This index may appear in fields where the relevant field in the relevant table of `ConceptNet 4` had a null entry (i.e. the empty string was the actual input).

Undefined Index = -2. This index is useful when an entry in a field refers to an object that does *not* actually appear in the appropriate table for that object. In other words, an index equal to -2 in a specific field implies that the index found originally for that field in the `ConceptNet 4` database was pointing to an object that did not actually exist in the database [1].

B.2 Files with the Tables of the Database

In this part we describe the tables that we derived from the original tables of the `ConceptNet 4` database.

Assertions

Table B.1 presents the format of each line in the file describing the assertions. Each line is composed of 14 integers separated by a space. Each line ends with a new line character `'\n'` right after the last integer. There are four indicators per assertion; in Table B.1 we have compressed them in one entry (the last entry) for clarity of presentation. These four indicators are, in that order, the *frame indicator*, the *surface form indicator*, the *raw assertion indicator*, and the *score indicator*.

Table B.1: The format of each line in the file describing the assertions. All the entries are integers separated by a space. Each line ends with a new line character `'\n'` right after the last integer. There are four indicators per assertion; the frame indicator, the surface form indicator, the raw assertion indicator, and the score indicator.

id	concept 1 index	concept 2 index	relation index	frequency index	best frame index	best surface 1 index	best surface 2 index	best raw assertion index	score	indicators

The first 2 lines of the file are shown below:

```
$ head -n 2 inputFiles/assertions/ConceptNet4Assertions.txt
2 0 1 5 0 0 0 1 0 1 0 0 0 0
```

[1] Actually, in the original tables of `ConceptNet 4` we find IDs instead of indices, but this is a more convenient description.

```
3 2 3 6 0 1 2 3 1 1 0 0 0 0
$
```

Number of Lines. The file has 566094 lines.

Permissible Values for each Field. The permissible values are described below.

id: The ID of each assertion in the original `ConceptNet 4` database. There are 566094 different values from the set $\{2, 3, \ldots, 898685\}$. This is the only set where not all integers are covered.

concept 1 index, concept 2 index: There are 279497 different concept IDs that appear in the assertions. Hence, the values come from the set $\{0, 1, \ldots, 279496\}$.

relation index: Integers from the set $\{0, 1, \ldots, 26\}$.

frequency index: Integers from the set $\{0, 1, \ldots, 10\}$.

best frame index: Integers from the set $\{-1\} \cup \{0, 1, \ldots, 2752\}$. Note that the frame can be null.

best surface 1 index, best surface 2 index: Integers from the set $\{-1\} \cup \{0, 1, \ldots, 375589\}$. Note that a surface form can be null.

best raw assertion index: Integers from the set $\{-2, -1\} \cup \{0, 1, \ldots, 525179\}$. Note that the raw assertion can be null *or undefined*.

score: Integers from the set $\{-10, -9, \ldots, 147\}$.

frame indicator: See Table 1.1.

surface form indicator: See Table 1.2.

raw assertion indicator: See Table 1.3.

score indicator: Integers from the set $\{0, 1, \ldots, 9\}$. See Table 1.8.

Concepts

Table B.2 presents the format of each line in the file describing the concepts. Each line is composed of one integer and a text description of the concept. These two are separated by a space. Hence, once we read an integer and a space, whatever remains until a new line character `'\n'` is encountered is the text description of the particular concept.

Table B.2: The format of each line in the file describing the concepts. Each line has two entries; a number and a string describing the concept. Each line ends with a new line character `'\n'` right at the end of the string describing the concept.

id	text

The first 2 lines of the file are shown below:

```
$ head -n 2 inputFiles/concepts/ConceptNet4Concepts.txt
5 something
6 to
$
```

Number of Lines. The file has 279885 lines.

Permissible Values for each Field.

id: Integers from the set $\{5, 6, \ldots, 482783\}$.

text: Longest string length is 204 characters (ID 211344).

Relations

Table B.3 presents the format of each line in the file describing the relations. Each line is composed of one integer and two strings describing the relation. Each field is separated by a space. Note that this does not leave any ambiguity among the strings, since the field `name` is a single word. Each line ends with a new line character `'\n'`.

Table B.3: The format of each line in the file describing the relations. Each line has three entries; a number and two strings. The first of the two strings (`name`) is a single word and the second string (`description`) is a more detailed description of the relation. Each line ends with a new line character `'\n'` right at the end of the second string.

id	name	description

The first 2 lines of the file are shown below:

```
$ head -n 2 inputFiles/relations/ConceptNet4Relations.txt
1 HasFirstSubevent What do you do first to accomplish it?
2 HasLastSubevent What do you do last to accomplish it?
$
```

Number of Lines. The file has 27 lines.

Permissible Values for each Field.

id: Integers from the set $\{1, 2, \ldots, 31\}$.

name: Longest string length is 21 characters (ID 12).

description: Longest string length is 50 characters (ID 28).

Frequencies

Table B.4 presents the format of each line in the file describing the frequencies. Each line is composed of two integers and one string describing the frequency. Each field is separated by a space. Note that this does not leave any ambiguity among the strings, since the field `name` is a single word. Each line ends with a new line character `'\n'`.

Table B.4: The format of each line in the file describing the frequencies. Each line has three entries; two numbers and one string. Each line ends with a new line character `'\n'`.

id	value	text

The first 2 lines of the file are shown below:

```
$ head -n 2 inputFiles/frequencies/ConceptNet4Frequencies.txt
1 5
2 7 often
$
```

Number of Lines. The file has 11 lines.

Permissible Values for each Field.

id: Integers from the set $\{1, 2, \ldots, 1403\}$.

value: Integers from the set $\{-10, -9, \ldots, 10\}$.

text: Longest string length is 13 characters (ID 3).

Frames

Table B.5 presents the format of each line in the file describing the frames. Each line is composed of three integers and one string describing the frame. Each field is separated by a space. Each line ends with a new line character `'\n'`.

Table B.5: The format of each line in the file describing the frames. Each line has four entries; three numbers and one string. Each line ends with a new line character `'\n'`.

id	relation index	frequency index	text

The first 2 lines of the file are shown below:

```
$ head -n 2 inputFiles/frames/ConceptNet4Frames.txt
3 5 0 Somewhere {1} can be is next {2}
4 6 0 You can use {1} to {2}
$
```

Number of Lines. The file has 2753 lines.

Permissible Values for each Field.

id: Integers from the set $\{3, 4, \ldots, 3831\}$.

relation index: Integers from the set $\{0, 1, \ldots, 26\}$.

frequency index: Integers from the set $\{0, 1, \ldots, 10\}$.

text: Longest string length is 131 characters (ID 2788).

Surface Forms

Table B.6 presents the format of each line in the file describing the surface forms. Each line is composed of two integers and one string describing the surface form. Each field is separated by a space. Each line ends with a new line character `'\n'`.

Table B.6: The format of each line in the file describing the surface forms. Each line has three entries; two numbers and one string. Each line ends with a new line character `'\n'`.

id	concept index	text

The first 2 lines of the file are shown below:

```
$ head -n 2 inputFiles/surfaceForms/ConceptNet4SurfaceForms.txt
5 0 something
6 1 to
$
```

Number of Lines. The file has 375590 lines.

Permissible Values for each Field.

id: Integers from the set $\{5, 6, \ldots, 580314\}$.

concept index: Integers from the set $\{0, 1, \ldots, 279884\}$.

text: Longest string length is 255 characters (IDs 286820).

Raw Assertions

Table B.7 presents the format of each line in the file describing the raw assertions. Each line is composed of seven integers separated by a space. Each line ends with a new line character '\n'.

Table B.7: The format of each line in the file describing the raw assertions. Each line has seven integers separated by a space. Each line ends with a new line character '\n'.

id	sentence index	assertion index	surface 1 index	surface 2 index	frame index	score

The first two lines of the file are shown below:

```
$ head -n 2 inputFiles/rawAssertions/ConceptNet4RawAssertions.txt
3 0 0 0 1 0 1
4 1 1 2 3 1 1
$
```

Number of Lines. The file has 525180 lines.

Permissible Values for each Field.

id: Integers from the set $\{3, 4, \ldots, 1277256\}$.

sentence index: Integers from the set $\{0, 2, \ldots, 525170\}$.

assertion index: Integers from the set $\{0, 1, \ldots, 566093\}$.

surface 1 index, surface 2 index: Integers from the set $\{0, 1, \ldots, 375589\}$.

frame index: Integers from the set $\{0, 1, \ldots, 2752\}$.

score: Integers from the set $\{-10, -9, \ldots 124\}$.

Table B.8: The format of each line in the file describing the sentences. Each line has three entries; two numbers and one string which is the actual sentence. Each line ends with a new line character `'\n'`.

id	score	text

Sentences

Table B.8 presents the format of each line in the file describing the sentences. Each line is composed of three integers and one string describing the frame. Each field is separated by a space. Each line ends with a new line character `'\n'`.

The last two lines of the file are shown below:

```
$ tail -n 2 inputFiles/sentences/ConceptNet4Sentences.txt
2608286 1 A cloned animal is made of D.N.A.
2608290 1 An U.F.O, is made of alien material.
$
```

Number of Lines. The file has 525171 lines.

Permissible Values for each Field.

id: Integers from the set $\{715991, 715992, \ldots, 2608290\}$.

score: Integers from the set $\{-10, -9, \ldots, 48\}$.

text: The length of the largest string is 1216 characters (ID 1023955).

B.3 Mapping From ConceptNet 4

Here we describe the structure of the files that map IDs for various objects from `ConceptNet 4` to the IDs that we use for input. All the files have one integer per line. These integers refer to the *indices* in the appropriate table where the objects can be found. Hence, valid indices are non-negative integers. However, we may encounter either a null index (-1), or an undefined index (-2). Null indices indicate that there was no object with such an ID in the original `ConceptNet 4` database. On the other hand, undefined indices indicate that there was an object with such an ID in the original `ConceptNet 4` database, but it turns out that this object does not appear in the closure of the input defined by the assertions of the English language.

Map Assertion IDs From ConceptNet 4

We can see the first 10 lines below:

```
$ head -n 10 inputFiles/assertions/MapAssertionIDsFromConceptNet4.txt
-1
-1
0
1
-1
-1
2
3
4
5
```

```
$
```

Hence, there are no assertions with IDs $0, 1, 4$ and 5 in the `ConceptNet 4` database when we restrict the search in the set $\{0, 1, \ldots, 9\}$. On the other hand assertion with ID 2 appears in index 0 of the table of assertions, assertion with ID 3 appears in index 1 in the table of assertions, and so on.

Number of Lines. The file has 898686 lines.

Map Concept IDs From ConceptNet 4

We can see the first 10 lines below:

```
$ head -n 10 inputFiles/concepts/MapConceptIDsFromConceptNet4.txt
-1
-1
-1
-1
-1
0
1
2
3
4
$
```

Hence, there are no concepts with IDs $0, 1, \ldots, 4$ in the `ConceptNet 4` database when we restrict the search in the set $\{0, 1, \ldots, 9\}$. On the other hand concept with ID 5 appears in index 0 of the table of concepts, concept with ID 6 appears in index 1 in the table of concepts, and so on.

Number of Lines. The file has 482784 lines.

Map Relation IDs From ConceptNet 4

We can see the first 10 lines below:

```
$ head -n 10 inputFiles/relations/MapRelationIDsFromConceptNet4.txt
-1
0
1
2
3
4
5
6
7
8
$
```

Hence, there is no relation with ID 0 in the `ConceptNet 4` database when we restrict the search in the set $\{0, 1, \ldots, 9\}$. On the other hand relation with ID 1 appears in index 0 of the table of relations, relation with ID 2 appears in index 1 in the table of relations, and so on.

Number of Lines. The file has 32 lines.

Map Frequency IDs From ConceptNet 4

We can see the first 10 lines below:

```
$ head -n 10 inputFiles/frequencies/MapFrequencyIDsFromConceptNet4.txt
-1
0
1
2
-1
-1
-1
-1
-1
-1
$
```

Hence, when we restrict the search in the set $\{0, 1, \ldots, 9\}$, only the IDs 1, 2, 3 actually appear in the original `ConceptNet 4` database and these are mapped respectively to indices $0, 1$, and 2. All the other frequency IDs are invalid (-1) in that region.

Number of Lines. The file has 1404 lines.

Map Frame IDs From ConceptNet 4

We can see the first 10 lines below:

```
$ head -n 10 inputFiles/frames/MapFrameIDsFromConceptNet4.txt
-1
-1
-1
0
1
2
3
4
5
6
$
```

Hence, there are no frames with IDs $0, 1$, and 2 in the `ConceptNet 4` database when we restrict the search in the set $\{0, 1, \ldots, 9\}$. On the other hand frame with ID 3 appears in index 0 of the table of frames, frame with ID 4 appears in index 1 in the table of frames, and so on.

Number of Lines. The file has 3832 lines.

Map Surface Form IDs From ConceptNet 4

We can see the first 10 lines below:

```
$ head -n 10 inputFiles/surfaceForms/MapSurfaceFormIDsFromConceptNet4.txt
-1
-1
```

```
-1
-1
-1
0
1
2
3
4
$
```

Hence, there are no surface forms with IDs $0, 1, \dots, 3$, and 4 in the `ConceptNet 4` database when we restrict the search in the set $\{0, 1, \dots, 9\}$. On the other hand surface form with ID 5 appears in index 0 of the table of surface forms, surface form with ID 6 appears in index 1 in the table of surface forms, and so on.

Number of Lines. The file has 580315 lines.

Map Raw Assertion IDs From ConceptNet 4

We can see the first 10 lines below:

```
$ head -n 10 inputFiles/rawAssertions/MapRawAssertionIDsFromConceptNet4.txt
-1
-1
-1
0
1
-1
-1
2
3
4
$
```

Hence, there are no raw assertions with IDs $0, 1, 2, 5$, and 6 in the `ConceptNet 4` database when we restrict the search in the set $\{0, 1, \dots, 9\}$. On the other hand raw assertion with ID 3 appears in index 0 of the table of raw assertions, raw assertion with ID 4 appears in index 1 in the table of raw assertions, and so on.

Number of Lines. The file has 1277257 lines.

Map Sentence IDs From ConceptNet 4

We can see the last 10 lines below:

```
$ tail -n 10 inputFiles/sentences/MapSentenceIDsFromConceptNet4.txt
525164
525165
525166
525167
525168
525169
-2
-2
-2
```

```
525170
$
```

Hence, *there are* sentences with IDs 2608287, 2608288, 2608289 in the `ConceptNet 4` database when we restrict the search in the set $\{2608281, 2608282, \ldots, 2608290\}$. However, it turns out that these sentences are *not* referenced by any raw assertion which appears in the database. Recall that we include only those raw assertions that appear as *best* raw assertions for at least one assertion in the English language.

On the other hand sentences with IDs $2608281, 2608282, \ldots, 2608286$ appear in indices $525164, 525165, \ldots, 525169$ of the table of sentences. Moreover, sentence with ID 2608290 is mapped to index 525170 in the table of sentences.

Number of Lines. The file has 2608291 lines.

B.4 Mapping To ConceptNet 4

No file is needed for this direction. Every data structure has an entry `id` that stores the integer of the actual ID for that particular object found in `ConceptNet 4`.

B.5 Lists of Edges: Directed and Undirected Multigraph

The same file is used both for the directed multigraph as well as the undirected multigraph. Table B.9 shows the structure of the file.

Table B.9: The structure of the file with the edges in the case of the directed and undirected multigraph.

concept 1 index	concept 2 index	assertion index

The first ten lines of the file are shown below.

```
$ head -n 10 inputFiles/edges/ConceptNet4EdgesDM.txt
0 1 0
2 3 1
7 3 2
8 9 3
10 3 4
11 203359 5
12 13 6
100569 15 7
46006 20 8
22 203360 9
$
```

Hence, the first edge is between concepts with indices (*not IDs*) 0 and 1 and the assertion justifying that edge has index 0 (the very first one). The second edge is between concepts with indices (*not IDs*) 2 and 3 and the assertion justifying that edge has index 1, and so on for the rest.

Number of Lines. The file has 566094 lines.

Table B.10: The structure of the file with the edges in the case of the directed graph.

concept 1 index	concept 2 index	number of assertions	assertion 1 index	assertion 2 index	...	assertion N index

B.6 Lists of Edges: Directed Graph

Table B.10 shows the structure of the file with the edges of the directed graph.

The first ten lines of the file are shown below.

```
$ head -n 10 inputFiles/edges/ConceptNet4EdgesDG.txt
0 0 2 102882 691
0 1 1 0
0 3 1 176972
0 4 1 14259
0 6 1 42755
0 7 1 31529
0 11 1 29344
0 13 1 161947
0 14 1 144144
0 15 1 35915
$
```

Hence, the first edge is a self loop for the concept with index (*not ID!*) 0 and there are two assertions justifying that loop; those with indices 102882 and 691. The second edge is an edge between the concepts with indices (*again, not IDs!*) 0 and 1, and there is one assertion justifying that edge which has index 0. Similarly for the rest.

Number of Lines. The file has 478929 lines.

B.7 Lists of Edges: Undirected Graph

Table B.11 shows the structure of the file with the edges of the directed graph.

Table B.11: The structure of the file with the edges in the case of the undirected graph.

concept 1 index	concept 2 index	number of assertions	assertion 1 index	assertion 2 index	...	assertion N index

The first ten lines of the file are shown below.

```
$ head -n 10 inputFiles/edges/ConceptNet4EdgesUG.txt
0 0 2 102882 691
0 1 1 0
0 3 1 176972
0 4 5 14259 338737 174978 192462 156888
0 6 1 42755
0 7 1 31529
0 11 1 29344
0 13 1 161947
```

```
0 14 1 144144
0 15 1 35915
$
```

Hence, the first edge is a self loop for the concept with index (*not ID!*) 0 and there are two assertions justifying that loop; those with indices 102882 and 691. The second edge is an edge between the concepts with indices (*again, not IDs!*) 0 and 1, and there is one assertion justifying that edge which has index 0. Similarly for the rest. Note that this time the fourth edge (i.e. has index 3) between concepts with indices 0 and 4 is justified by 5 assertions as opposed to the 1 assertion shown in Table B.10. The reason is that for the undirected edges we can not distinguish between the source and the destination of the edge as there is no orientation on the edge. Hence, when we write down the undirected edges, the convention that we follow is that we write first the node (concept) with smallest index, and second the node (concept) with largest index.

Number of Lines. The file has 465072 lines.

Appendix C

Directory Structure, Timestamps and File Sizes

Below we see the directory structure, creation time and date for each file, and finally the size of each file.

```
$ ls -lR inputFiles/
total 0
drwxr-xr-x  2 user  staff  136 Oct 24 12:13 assertions/
drwxr-xr-x  2 user  staff  136 Oct 24 12:13 concepts/
drwxr-xr-x  2 user  staff  170 Oct 24 12:13 edges/
drwxr-xr-x  2 user  staff  136 Oct 24 12:13 frames/
drwxr-xr-x  2 user  staff  136 Oct 24 12:13 frequencies/
drwxr-xr-x  2 user  staff  136 Oct 24 12:13 rawAssertions/
drwxr-xr-x  2 user  staff  136 Oct 24 12:13 relations/
drwxr-xr-x  2 user  staff  136 Oct 24 12:13 sentences/
drwxr-xr-x  2 user  staff  136 Oct 24 12:13 surfaceForms/

inputFiles//assertions:
total 69744
-rw-r--r--  1 user  staff  30858317 Oct 24 12:13 ConceptNet4Assertions.txt
-rw-r--r--  1 user  staff   4849324 Oct 24 12:13 MapAssertionIDsFromConceptNet4.txt

inputFiles//concepts:
total 17048
-rw-r--r--  1 user  staff  6270698 Oct 24 12:13 ConceptNet4Concepts.txt
-rw-r--r--  1 user  staff  2456782 Oct 24 12:13 MapConceptIDsFromConceptNet4.txt

inputFiles//edges:
total 59432
-rw-r--r--  1 user  staff  10200385 Oct 24 12:13 ConceptNet4EdgesDG.txt
-rw-r--r--  1 user  staff  10188521 Oct 24 12:13 ConceptNet4EdgesDM.txt
-rw-r--r--  1 user  staff  10031293 Oct 24 12:13 ConceptNet4EdgesUG.txt

inputFiles//frames:
total 200
-rw-r--r--  1 user  staff  85312 Oct 24 12:13 ConceptNet4Frames.txt
-rw-r--r--  1 user  staff  15892 Oct 24 12:13 MapFrameIDsFromConceptNet4.txt

inputFiles//frequencies:
total 24
-rw-r--r--  1 user  staff   155 Oct 24 12:13 ConceptNet4Frequencies.txt
-rw-r--r--  1 user  staff  4202 Oct 24 12:13 MapFrequencyIDsFromConceptNet4.txt
```

```
inputFiles//rawAssertions:
total 50208
-rw-r--r--  1 user  staff  19879266 Oct 24 12:13 ConceptNet4RawAssertions.txt
-rw-r--r--  1 user  staff   5821381 Oct 24 12:13 MapRawAssertionIDsFromConceptNet4.txt

inputFiles//relations:
total 16
-rw-r--r--  1 user  staff  1028 Oct 24 12:13 ConceptNet4Relations.txt
-rw-r--r--  1 user  staff    86 Oct 24 12:13 MapRelationIDsFromConceptNet4.txt

inputFiles//sentences:
total 69992
-rw-r--r--  1 user  staff  26015887 Oct 24 12:13 ConceptNet4Sentences.txt
-rw-r--r--  1 user  staff   9814447 Oct 24 12:13 MapSentenceIDsFromConceptNet4.txt

inputFiles//surfaceForms:
total 29112
-rw-r--r--  1 user  staff  11768624 Oct 24 12:13 ConceptNet4SurfaceForms.txt
-rw-r--r--  1 user  staff   3132195 Oct 24 12:13 MapSurfaceFormIDsFromConceptNet4.txt
$
```

Appendix D

Further Issues with the Database

In this appendix we describe further issues that we have observed on `ConceptNet 4` but did not appear during the derivation process of the input files.

D.1 num_assertions on conceptnet_concept

In theory the entries found in that column of the table conceptnet_concept could be used in order to calculate the degree of the node (concept) in the induced directed multigraph. However, this is not the case.

Let us take the very first concept that has ID equal to 5 and first of all ignore the scores. The first line below indicates that concept with ID 5 does not appear among assertions that are not in the English language. This way we do not have to restrict the language being English in further SQL queries.

```
sqlite> select count(*) from conceptnet_assertion where
   ...> ((concept1_id = 5) or (concept2_id = 5)) and (language_id is not 'en');
0
sqlite> select count(id) from conceptnet_assertion where (concept1_id = 5);
2816
sqlite> select count(id) from conceptnet_assertion where (concept2_id = 5);
147
sqlite> select count(id) from conceptnet_assertion where
   ...> (concept1_id = 5) and (concept2_id = 5);
2
sqlite> select count(id) from conceptnet_assertion where
   ...> (concept1_id = 5) or (concept2_id = 5);
2961
```

Note that $2961 = 2816 + 147 - 2$. However, all these numbers are different from 2887 which is the entry in the num_assertions column of the table conceptnet_concept. When we restrict to scores being positive, we still can not justify the numbers.

```
sqlite> select count(id) from conceptnet_assertion where
   ...> (concept1_id = 5) and (score > 0);
2754
sqlite> select count(id) from conceptnet_assertion where
   ...> (concept2_id = 5) and (score > 0);
139
sqlite> select count(id) from conceptnet_assertion where
   ...> (concept1_id = 5) and (concept2_id = 5) and (score > 0);
2
sqlite> select count(id) from conceptnet_assertion where
```

```
   ...> ((concept1_id = 5) or (concept2_id = 5)) and (score > 0);
2891
```